

Interfaces Gráficas en Matlab

Usando GUIDE

Evento:

CONATEC 2002

Sede:

INSTITUTO TECNOLÓGICO DE CIUDAD MADERO

Instructor:
M. C. José Jaime Esqueda Elizondo
Universidad Autónoma de Baja California, Unidad Tijuana

Daniel de León Cárdenas
Jorge Espinosa Caballero
José Luis Vargas Cruz

Noviembre de 2002

Introducción

GUIDE (**Graphical User Interfase Development Environment**) es un juego de herramientas que se extiende por completo el soporte de MATLAB, diseñadas para crear GUIs (Graphical User Interfaces) fácil y rápidamente dando auxiliando en el diseño y presentacion de los controles de la interfaz, reduciendo la labor al grado de seleccionar, tirar, arrastrar y personalizar propiedades.

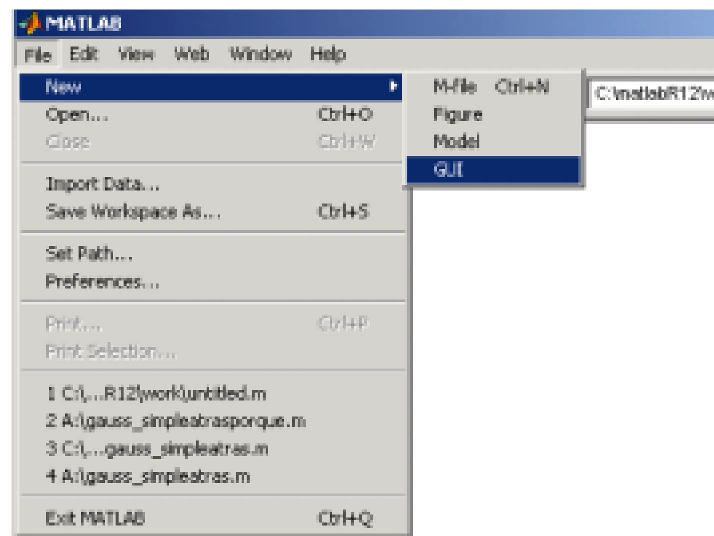
Una vez que los controles están en posición se editan las funciones de llamada (Callback) de cada uno de ellos, escribiendo el código de MATLAB que se ejecutará cuando el control sea utilizado. Siempre será difícil diseñar GUIs, pero no debería ser difícil implementarlas. GUIDE esta diseñado para ser menos tediosos el proceso de aplicación de la interfaz grafica y obviamente para trabajar como herramienta de trazado de GUIs, entre sus poderosos componentes esta el **editor de propiedades (property editor)**, este se encuentra disponible cualquier momento que se esté lidiando con los controles de MATLAB, el editor de propiedades por separado se puede concebir como una herramienta de trazado, y asistente de codificación (revisión de nombres y valores de propiedades). Cuando se fusiona con el panel de control, el editor de menú, y herramienta de alineación, resulta una combinación que brinda inigualable control de los gráficos en MATLAB.

Utilizando GUIDE

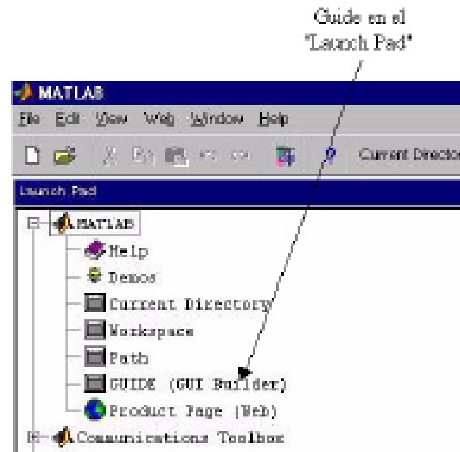
A la herramienta GUIDE se accede de varias maneras, la primera de ellas es tecleando guide en la ventana de comando.

```
>> guide
```

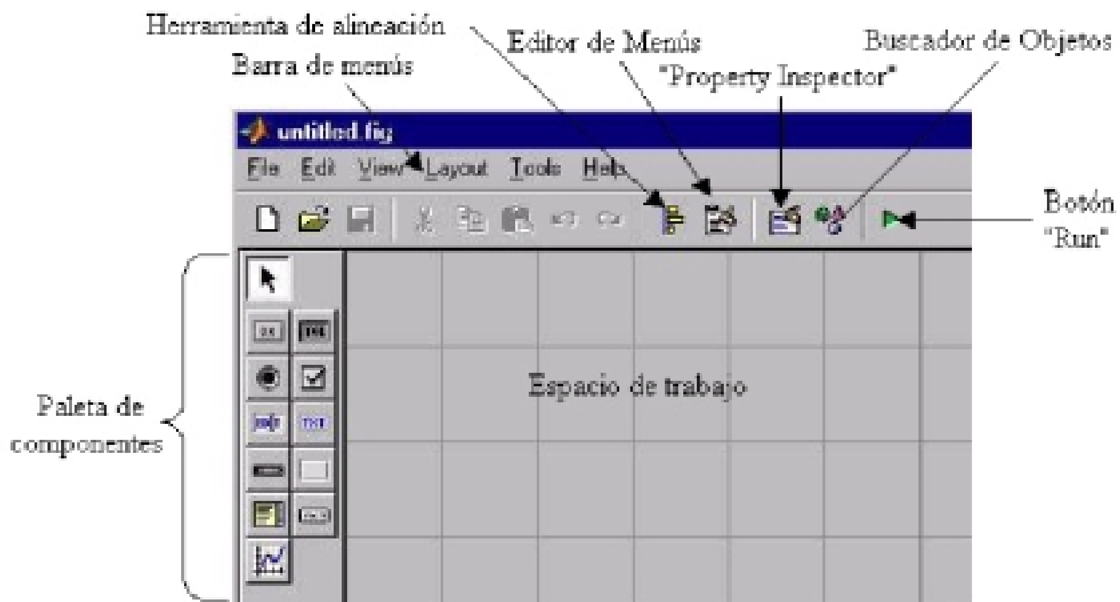
Otra manera de entrar a GUIDE es través del File opción New y por ultimo el GUI, (como se muestra en la figura).



Otra de ellas consiste en buscar en el "Launch Pad" la opción referente a Matlab, hacer clic en ella, con lo que Matlab despliega las opciones que contiene, entre las que se encuentra la opción "Guide (GUI Builder)", tal como se muestra en la figura.



La ventana principal de GUIDE es la siguiente:



Las Componentes principales de GUIDE son:

Barra de Menús: Aquí se encuentran las funciones elementales de Edición de GUI’s.

Paleta de Componentes (component Palette): Aquí se encuentras los uicontrols estos componentes permite seleccionar los controles (objetos) que son los que se muestra en la figura.

La Barra de Herramienta: En ella se encuentran lo siguientes botones

Botón de ejecución (Run button): Al presionarse de crea la figura de la interfaz diseñada en el Layout Área.



Alineación de Componentes (Alignment tool): esta opción permite alinear los componentes que se encuentra en el área de trabajo (Layout Área) de manera personalizada.



Propiedades del Inspector (Property Inspector): con esta opción se asignan y modifican las propiedades de cada objeto en forma personalizada.



Navegador de Objetos (Object Browser): Muestra todos los objetos que se encuentra en la figura (en forma de árbol) y a través de Object Browser se puede seleccionar los objetos.

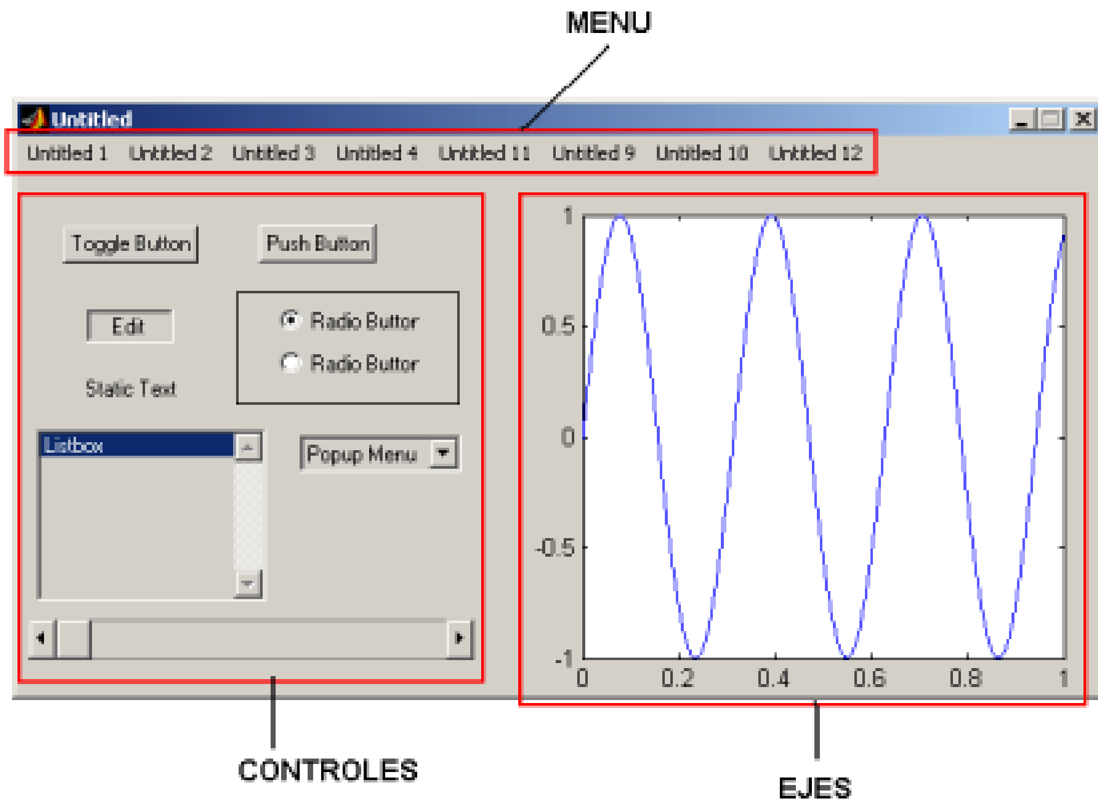


Editor de Menús (Menú Editor): El redactor de Menú crea menús de ventana y menús de contexto.



La Interfaz de Grafica de Usuario (GUI) se crea en una ventana de figura que consta de los siguientes componentes:

1. Menú de interfaz con el usuario.
2. Dispositivo de control de interfaz con el usuario.
3. Ejes para exhibir graficas o imágenes



`Puede personalizar el GUIDE en la opción` *`Preferences`* `hallada en el menú` *`File,`* `ahi posible desplegar los nombres de los componentes hallados en la paleta y la de presentar las herramientas.`

FLUJO DE OPERACIÓN CON GUI

Con una GUI, el flujo de computo esta controlado por las acciones en la interfaz. Mientras que en un guión el flujo de comandos esta predeterminado, el flujo de operaciones con una GUI no lo está. Los comandos para crear una interfaz con el usuario se escribe en un guión, la interfaz imboca el guión que se ejecute, mientras la interfaz del usuario permanece en la pantalla aunque no se haya completado la ejecución del guión.

En la figura 1 se muestra el concepto básico de la operación del software con una GUI. Cuando se interactua con un control, el programa registra el valor de esa opción y ejecuta los comandos prescritas en la cadena de invocación. Los menús de interfaz con el usuario, los botones, los menús desplegables, los controladores deslizantes y el texto editable son dispositivos que controlan las operaciones del software. Al completarse la ejecución de las instrucciones de la cadena de invocación, el control vuelve a la interfaz para que puedan elegirse otra opción del menú. Este ciclo se repite hasta que se cierra la GUI.

El control guarda un string que describe la accion a realizar cuando se invoca puede consistir en un solo comando de MATLAB o una secuencia de comandos, o en una llamada a una función. Es recomendable utilizar llamadas a funciones, sobre todo cuando se requieren de mas de unos cuantos comandos en la invocación.

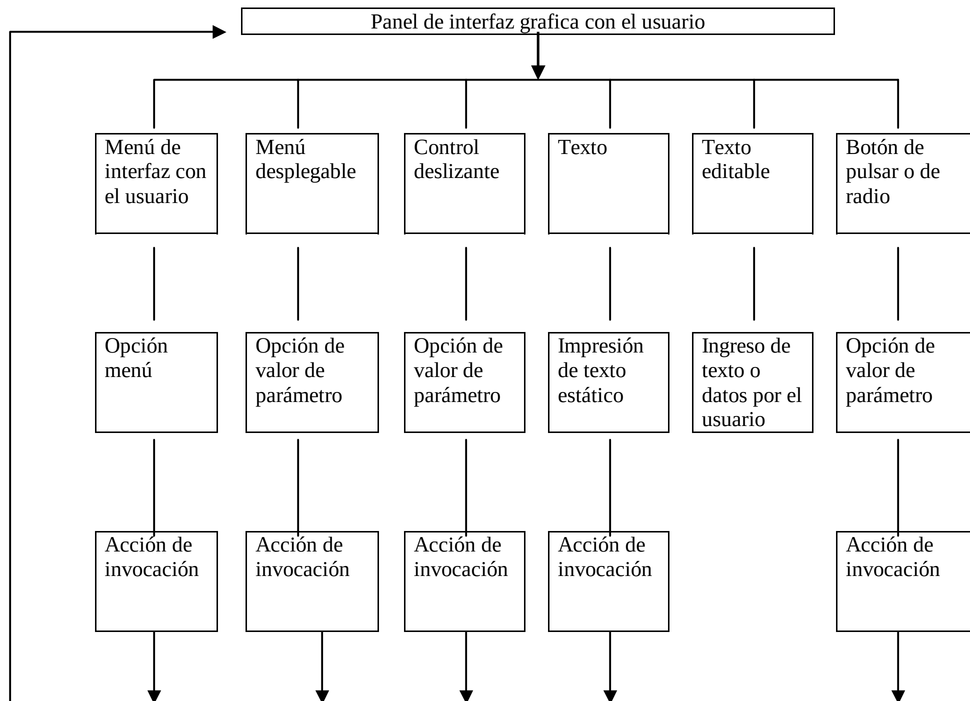
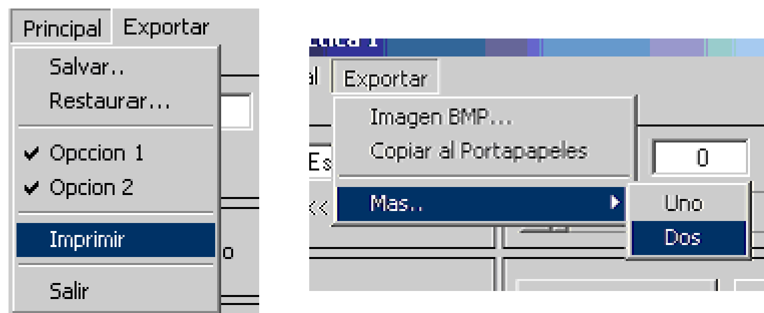


DIAGRAMA 1 Flujo de opciones de GUI

Basicamente solo se necesita entender cinco comandos para poder describir una GUI: `uimenu`, `uicontrol`, `get`, `set` y `axes`. No obstante, lo que hace relativamente complicadas a estos comandos es el gran numero de formas de uso que tiente. Es imposible describir todos lo tipo de situaciones, pues requiere demasiado espacio y seria muy laborioso leerlo. Por tanto, solo se trataremos de explicar los elementos básicos de una GUI a través de ejemplos. Los lectores que deseen información mas detallada sobre los comandos consultar MATLAB: building a graphical user interface y al final del manual encontraran un apendice que describe las propiedades de los controles.

MENU DE INTERFAZ CON EL USUARIO

El menú de interfaz con el usuario es un menú o un grupo de menús que se encuentran en la parte superior de una ventana de la figura. El menú se desarrolla de arriba hacia abajo cuando se hace clic con el ratón y se muestra una lista de opciones siguiente figura. Cuando se elige una opción de la lista, es posible que se desenrolle otro nivel de menús (si el menú se diseño para ello).



Los menus de interfaz con el usuario se especifican con uimenu, cuya sintaxis es la siguiente:

```
m1 = uimenu (gcf, ...
        'Label', 'cadena de rótulos 1', ...
        'Position' [números de prioridad (entero)], ...
        'Backgroundcolor' [r, g, b], ...
        'CallBack' ' cadena de invocación')

m2 = uimenu (gcf, ...
        'Label', 'cadena de rótulos 2', ...
        'Position' [números de prioridad (entero)], ...
        'Backgroundcolor' [r, g, b], ...
        'CallBack' ' cadena de invocación')

m3 = uimenu (gcf, ...
        'Label', 'cadena de rótulos 1', ...
        'Position' [números de prioridad (entero)], ...
        'Backgroundcolor' [r, g, b], ...
        'CallBack' ' cadena de invocación')
```

Aquí se esta suponiendo que hay tres menús en una ventana de figura. En los comandos, m1 m2, ... son mangos (get) de los menús que a menudo son necesarios en CallBack y otros comandos. Los argumentos que siguen a gcf son las propiedades del menú y tiene el siguiente significado:

a) 'Label', 'cadena de rótulos especifica el rotulo que aparecerá en el menú.
b) 'Position', ***k*** determina la posición secuencial del rotulo en el menú, donde k es un numero entero que indica en orden de prioridad.
c) 'Backgroundcolor', [r, g, b], especificas el color de segundo plano del menú.
d) 'CallBack' ' cadena de invocación' especifica los comandos que se ejecutan cuando se selecciona el rotulo.

En la sintaxis anterior es posible omitir las líneas correspondientes a positicion y backgroundcolor si son aceptable los valores por omisión. Además, callBack no es

necesario cuando el menú va seguido de una lista de opciones que se abre cuando se hace clic en el menú.

También podemos especificar otras opciones; sin embargo, la mejor forma de aprenderlas es ejecutando el comando get (mango) después de ejecutarse el comando uimenu, donde mango deber ser como m1 o m2 de la explicación de la sintaxis anterior. El comando get (mango) devuelve las propiedades actuales del menú, que en su mayoría se establecen por omisión pero que pueden modificarse en los argumentos del enunciado del comando uimenu.

Si una opción del menú tiene subopciones, estas también se especifican como uimenu. Si consideramos la lista de opciones para el primer menú cuyo mango es m1, la sintaxis con tres opciones seria la siguiente:

```
m1sA = uimenu (m1, ...
        'Label', 'opcion A', ...
        'CallBack' ' cadena de invocación')

m1sB = uimenu (m1, ...
        'Label', 'opcion B', ...
        'CallBack' ' cadena de invocación')

m1sC = uimenu (m1, ...
         'Label', 'opcion C', ...
        'CallBack' ' cadena de invocación')
```

Las tres opciones pertenecen al primer menú con mango m1. se omitieron as propiedades de Position y BackgroundColor, pero pueden incluirse si se desea.

La propiedad CallBack es importante aquí si uicontrol es para el nivel terminal del menú. La cadena de invocación es una cadena que consiste en un comando, un conjunto de ordenes o una llamada de función. En esta cadena se especifican todas las tareas de computo que deben ejecutarse al elegirse la opción. Las instrucciones pueden ser un solo comando o varios; sin embargo, a fin de simplificar la programación es recomendable no escribir mas de unos cuantos comandos o más de una llamada de función en la cadena de invocación. Todos los detalles de los cálculos se pueden escribir en un archivo M.

Nota: el archivo M puede o no ser de función. En cualquier caso, 'cadena de invocación' es remplazado por 'nombre del archivo M' la diferencia entre utilizar un archivo M de función y uno no de función es la siguiente. Si se emplea un archivo M de función, es necesario proporcionar como argumentos las variables necesarias para el archivo M de función, pues de lo contrario este no podría ver las variables del archivo M que invoca. Por otro lado, si se utiliza un archivo M no de función, todas las variables del archivo M invocado estarán visibles.

CONTROLES DE LA INTERFAZ DE GUIDE.

Los controles de la interfaz con el usuario en MATLAB se especifican con la orden uicontrol. Estos controles tienen mucho en común con los menús de la interfaz con el usuario, pero los primeros tienen mucho estilos. La sintaxis de uicontrol es

```
k = uicontrol ('Style', 'especificacoin de estilo', …
                'String', 'cadena para exhibir', …
                'Value', [valor], …
                'BackgroundColor', [r,g,b], …
                'Max' [valor], …
                'Min' [valor], …
                'Position', [izq, base, ancho, alto], …
                'CallBack', 'cadena de invocacion')
```

donde `'especificación de estilo'` es una de las siguientes cadenas:

`popup`
`push`
`radio`
`checkbox`
`slider`
`edit` **(texto editable)**
`text` **(texto estático)**
`frame`

Las propiedades de `iucontrol` son similares a las de `uimenu`. las propiedades que aparecen aquí por primera vez son:

a) 'Value', valor: especifica el valor por omisión de ajuste. En el caso de interruptores de encendido/apagado, valor es 0 o 1. En el caso de un control deslizante (`slider`), puede ser cualquier valor entre el mínimo y el máximo.
b) 'Min', Valor: establece el valor mínimo. Su significado difiere dependiendo del estilo.
c) 'Max', Valor: establece el valor máximo. Su significado difiere dependiendo del estilo.

Hay muchas más propiedades que pueden incluirse en los comandos de uicontrol, tal como sucede con las propiedades de uimenu, aunque al programar conviene minimizar el número de propiedades a fin de simplificar el guión. Si se desea saber mas acerca de las propiedades adicionales, investigue utilizando el comando `get`.

TXT Static Text **Texto estático. Un static text** puede exhibir símbolos, mensajes o incluso valores numéricos de una GUI, y puede colocarse en lugar deseado. El texto

estático no tiene cadenas de invocación. A continuación mostramos un ejemplo de texto estático.

```
k1 = uicontrol (´Style´, ´text´, …
                 ´String´, ´cadena para exhibir´, …
                 ´Position´, [20, 50, 140, 30])
```

El contenido de un texto exhibido puede modificarse si es necesario. Esto se hace con el comando set. Por ejemplo, ejecute el comando que sigue desde la ventana de comandos mientras está vigente el ejemplo anterior de orden uicontrol:

`set (k1, ´string´,` ´Ahora aparece un texto modificado.´)



Menú desplegable. Los pop-up menús difieren de los menús de interfaz con el usuario en que pueden aparecer en cualquier punto de la ventana de figura, mientras que los menús de interfaz con el usuario solo se localizan en la parte superior.



Los Push button generan una acción cuando das click con el puntero del ratón sobre ellos. Cuando usted da click en un push button, aparece presionado; Cuando sueltas el botón del ratón, el botón aparece levantado; y su rutina de llamada se ejecuta.



Casilla de verificación. Las casillas de verificación están diseñadas para realizar operaciones de encendido/apagado. La casilla activa o desactiva la aparición de los ejes. Las posiciones de encendido/apagado se registran en `Value` que puede examinarse con `get(handle, ´value´)`. Los comandos `axis on` y `axis off` se escriben en la cadena de invocación.



Botón de radio. Cuando solo se usa un botón de radio, no existe diferencia funcional alguna con respecto a una casilla de verificación. Por otro lado, los botones de radio en grupo son mutuamente exclusivos (es decir, si un botón esta encendido, todos los demás botones se apagan), mientras que las casillas de verificación son independientes entre sí. Sin embargo, esta características exclusiva de los botones de radio sólo puede implementarse mediante la programación del usuario en la cadena de invocación.



Barra deslizadora. Los sliders aceptan datos de entrada numéricos con un rango especifico. Los usuarios mueven la barra dejando presionado el botón

del mouse y arrastrándola, Dando click en el canal, en la flecha. La posición de la barra indica un valor numerico.



Texto editable. El dispositivo de texto editable permite al usuario teclear una cadena de entrada. Se pueden escribir varios valores numéricos en forma de vector o matriz como cadena mediante el mismo dispositivo; esta cadena se convertirá posteriormente en valores numéricos con el comando str2num.

Un ejemplo de uicontrol para texto editable es:

```
ed1 = uicontrol (gcf, ´Style´, ´edit´, …
                 ´Position´, [10, 260, 110, 20], …
                 ´CallBack´, inp_txt = get (ed1, ´´string´´)´)
```

Las palabras clave en el comando anterior son `´Style´, ´edit´` y `get (mango, ´string´)` que capturan el texto introducido.



Marcos. El estilo marcos puede servir para agrupar dispositivos como lo botones de radio o las casillas de verificación.



Botón de palanca. El toggle button genera una acción que indica un estado binario (on o off). Cuando das click en un toggle button , aparece presionado y permanece así hasta que sueltes el boton de el mouse, y en ese momento ejecuta la llamada. Un click del mouse subsecuente regresa al toggle button a su estado original y vuelve a ejecutar la rutina de llamada.



Cajas de lista. El componente List Box muestra una lista de artículos y permite a usuarios seleccionar unos o más artículos.

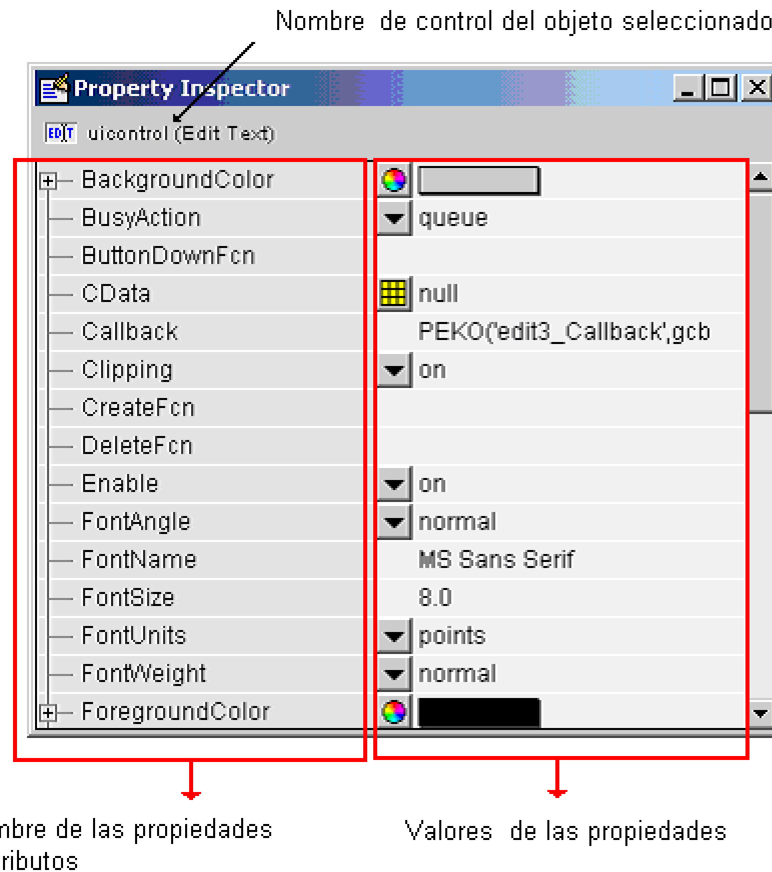
MÚLTIPLES EJES

Al crear una interfaz con el usuario, a menudo hay necesidad de trazar una o más gráficas dentro de la interfaz. Podemos usar el comando `subplot` para este fin, pero el comando `axes` es más flexible y ofrece opciones versátiles a los programadores.

El comando `axes` abre un eje en un punto especificado dentro de una ventana de figura. Aunque podemos abrir varios ejes en una ventana de figura con `axes`, primero consideramos únicamente uno.

Property Inspector

El inspector de propiedades esta compuesta de la siguiente forma como se muesta en la figura.



Cada uno de los controles tienen propiedades particulares, esta se pueden consultar en el apendice.

Practicando con GUIDE

Una vez ya conocidas las herramientas de GUIDE es conveniente que empecemos a usarlas. Como primera práctica utilizaremos todos los componentes, los haremos interactuar unos con los otros de tal manera que al usar un control, se produzca un efecto en los otros.

Practica 1. ***"Usando Todos los Controles".***

1.- Emboquemos la herramienta GUIDE. Anteriormente se mostraron las maneras de hacerlo, o bien solo escriba ***guide*** en la línea de comandos.

2.- Una vez en la herramienta lo primero que se hace es tirar en la área de trabajo dos controles, un Edit Text y Uno Estatic Text.

3.- Haga doble clic sobre el control Edit Text y Aparecera el Propety Inspector. En el modifique el atributo de color de fondo a Blanco. BackgroudColor puede modificarlo de dos maneras una es presionando y la otra es extendiendo las opciones interiores de la propiedad presionando en el signo de +, y colocando 1's en red, green y blue.

4.- Cambie el Texto por defecto del control Edit Text por: "Escriba Aquí".

Propiedad: "String"   Nuevo Valor: "Escriba Aquí"

5.- Haga mas ancho el control.

Coloque el mouse sobre uno de los rectángulos del control y estirelo.
O También:
Propiedad: "Position → Width"   Nuevo Valor: "20".

6.- Modifique es texto del control Static Text, A "<<Mensaje>>" y estirelo.

Propiedad: "String"   Nuevo Valor: "<<Mensaje>>"
Propiedad: "Position → Width"   Nuevo Valor: "20".

7.- Ejecute su GUI.

Presione y como no se ha grabado la interfaz, aparecerá un dialogo avisándole que no lo ha gravado, gravelo y entonces se ejecutara la GUI.

Contemple su interfaz, deberá lucir similar a la siguiente figura y le recomendamos cerrar la figura antes de hacer alguna modificación en la figura o en el archivo-M.



8.- Ahora Crearemos una función que al presionar Enter o una vez ya activo el control y presionemos fuera de el, se llame. Este evento invocará a la función "Callback". Coloque el mouse sobre el control, presione el botón derecho, Expanda "View Callbacks" y escoja "Callback". Entonces aparecerá la ventana de edición de archivos-M.

Estará ya colocado en la línea de la función correspondiente al control. Puede verificar el nombre:

function varargout = edit1_Callback(h, eventdata, handles, varargin)

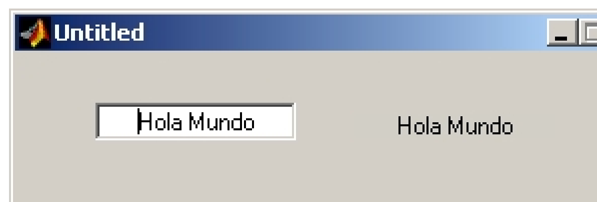
la sintaxis para nombrar las funciones es la siguiente:

function <variable salida> = ‘Nombre del control’_’Funcion llamada’(<manejador del objeto que llama>, <datos evento>, <manejador>, <variable entrada>)

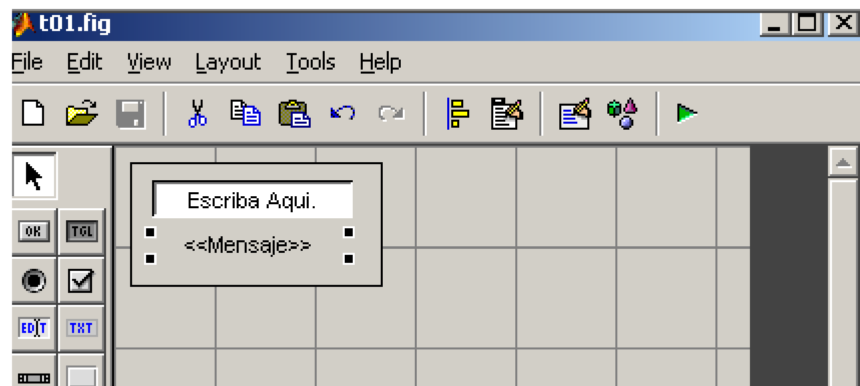
Escriba el siguiente código:

```
msg= get(h,'string')
set(handles.text1,'string',msg)
```

9.- Ejecute el Programa y Escriba en el Edit Text: “Hola Mundo” y presione enter.
Deberá tener el siguiente resultado.



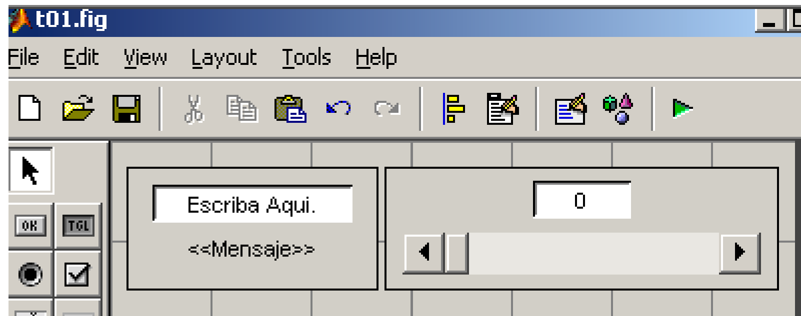
10.- Tire un Control Frame en la área de trabajo. Acomode los objetos de tal manera que luzcan de la siguiente manera.



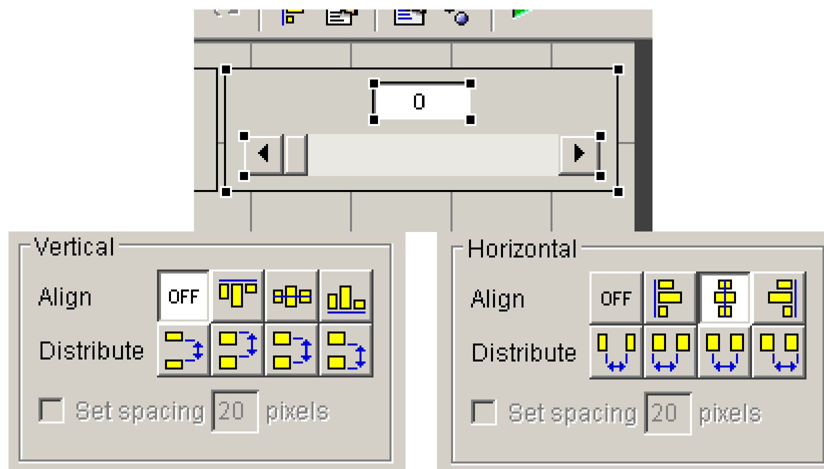
Nota: Tendrá un inconveniente, el Frame tapara los objetos anteriores, resuelva el problema presionando el botón derecho del mouse sobre el frame y seleccione “Send to Back”.

11.- Agregue al Área de Trabajo un Frame, un Slider y un edit box. En el Edit Box Cambie el Color de Fondo a Blanco.

Ordénelos de la Siguiente manera.



Nota: Para Hacer que luzcan ordenadamente los objetos le recomendamos alinearlos, a través de la herramienta Align Objects , seleccione los objetos y seleccione la configuración que se muestra:



12.- Agreguemos código que al deslizar el slider, el valor de este se presente en el edit box, y valor del edit box se represente en el slider al presionar sobre el edit box el botton derecho.

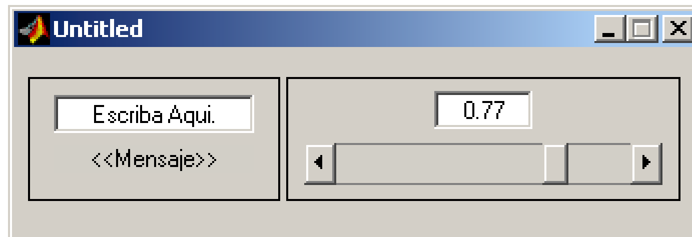
Valla a la función de llamado “CallBack” del slider y escriba el siguiente código:

```
value = get(h,'value');
set(handles.edit2,'string',value);
```

Vaya a la function de llamado “ButtonDownFcn” del edit text y y escriba el siguiente código:

```
set(handles.slider1,'value',str2num(get(h,'string')))
```

13.- Ejecute la Figura. Y Realice pruebas con ella.



14.- Supongamos que se requiere que el slider maneje un valor de 0 a 1000 y al momento de presionar las flechas de los costados el valor varié una unidad y al presionar entre el bloque deslizante y una flecha de los costados el valor varié 10 unidades. Para lograr estos efectos se requiere modificar las siguientes propiedades:

Propiedad: “Max”   Nuevo Valor: “1000”
Propiedad: “SliderStep → x”   Nuevo Valor: “0.001”
Propiedad: “SliderStep → y”   Nuevo Valor: “0.01”

X indica el % de cambio al presionar las flechas.
Y indica el % de cambio al presionar entre el bloque y la flecha de un costado.

Nota: Los valores en SliderStep son valores que se calculan a partir del porcentaje de cambio, como en x el cambio es de una milésima se indico el valor 0.001 y el cambio de y es de un centésima se indico el valor de 0.01.

15.- Agregue un Control Push Botton, y modifique la siguiente propiedad.

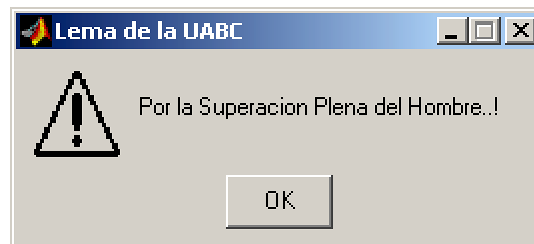
Propiedad: “String”   Nuevo Valor: “Lema”
Propiedad: “Position → Width”   Nuevo Valor: “15”

16.- Escriba en la función de llamado Callback del Push Botton, el siguiente código.

Warndlg ('Por la Superación Plena del Hombre..!','Lema de la UABC')

17.- Ejecute la GUI. Y Presione el botón.

Y se mostrara un dialogo de aviso con el lema de la UABC.

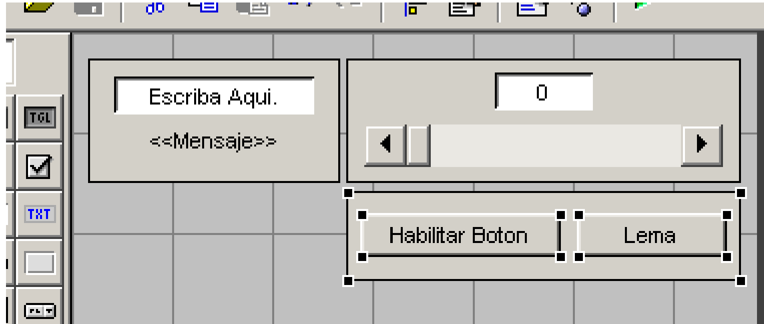


18.- Introduzca en la área de trabajo un Frame y un Toggle Button, modifique los siguientes atributos de los controles:

Control: togglebutton1
Propiedad: “String”   Nuevo Valor: “Habilitar Boton”
Propiedad: “Position → Width”   Nuevo Valor: “20”

Control: pushbutton1
Propiedad: “Enable”   Nuevo Valor: “off”

Ordene los controles para que luzcan así:



19.- Escriba el siguiente código en la función Callback del control togglebutton1.

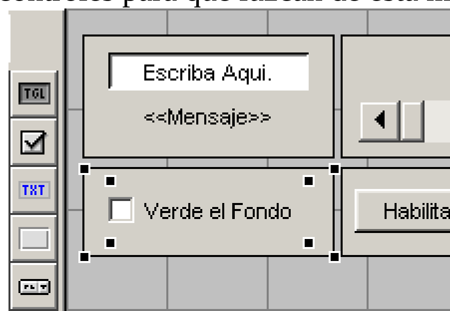
```
if get(h,'value') == 1
   set(handles.pushbutton1,'enable','on')
else
   set(handles.pushbutton1,'enable','off')
end
```

El objetivo de este código es habilitar o deshabilitar el control pushbutton1 dependiendo del estado de togglebutton1.

20.- Ejecute la figura. Y verifique que funcione.

21.- Agregue un Checkbox y un frame, y programemos el checkbox para cuando este marcado pinte el color de fondo. Modifique las propiedades del checkbox
Propiedad: “String”   Nuevo Valor: “Verde el Fondo”
Propiedad: “Position → Width”   Nuevo Valor: “20”

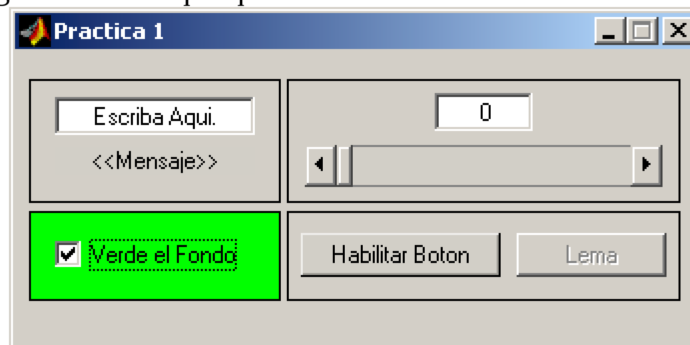
Entonces ordenemos a los controles para que luzcan de esta manera:



22.- Escribamos el siguiente código en la función de llamado Callback del control Checkbox1:

```
if get(h,'value') == 1
   set(h,'Backgroundcolor','green')
   set(handles.frame4,'Backgroundcolor','green')
else
   set(h,'Backgroundcolor','factory')
   set(handles.frame4,'Backgroundcolor','factory')
end
```

23.- Ejecute la figura. Y verifique que funcione.



24.- Ahora utilizaremos el control Popup Menu, Inserte un control Popup, Static Text y un frame. Cambie los atributos de los siguientes objetos:

Control: text2
Propiedad: “String”   Nuevo Valor: “Color del Fondo de la Pantalla” *
Propiedad: “Position → Width”   Nuevo Valor: “20”
Propiedad: “BackgroundColor → red, green, blue”   Nuevo Valor: “1, 1, 1”
*Escriba este valor en dos lineas con ayuda del editor de String del Property Inspector.

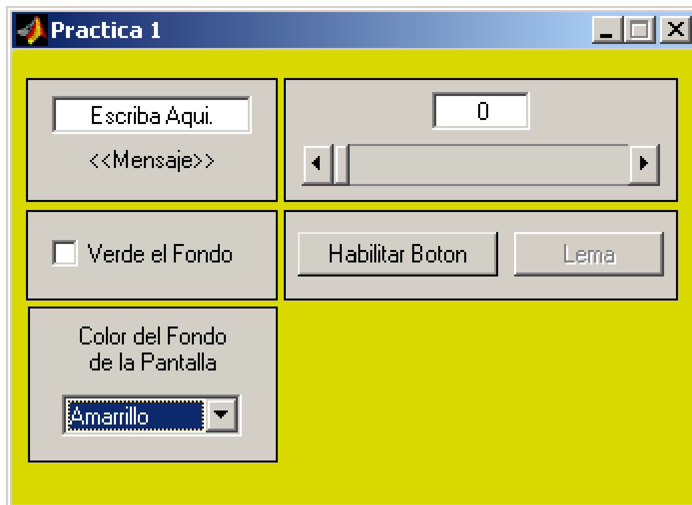
Control: popupmenu1
Propiedad: “String”   Nuevo Valor:
“Amarrillo|Morado|Cielo|Rojo|Verde|Azul|Blanco|Negro”

Ordene los controles para que luzcan de esta manera:



25.- En el Callback del Popup menú escriba lo siguiente:
color = {'y';'m';'c';'r';'g';'b';'w';'k'};
whitebg(char(color(get(h,'value'))))

26.- Ejecute la figura. Y verifique que funcione.



27.- Hará una lista de números selecionables a través de 3 Radios Buttons, donde cada Radio desplegara un conteo diferente el Listbox.

Inserte en el área de trabajo un Frame, un Listbox, un Static Text y 3 Radio Button. Modifique los siguientes atributos:

Control: text3
Propiedad: “String”   Nuevo Valor: “Contando..”

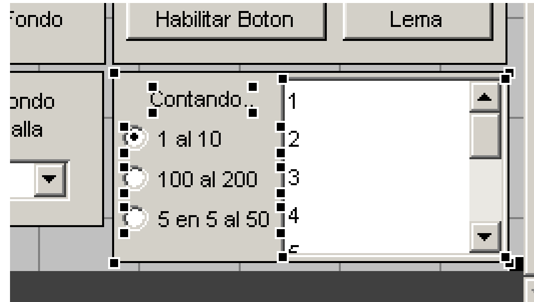
Control: radiobutton1
Propiedad: “String”   Nuevo Valor: “1 al 10”
Propiedad: “Value”   Nuevo Valor: “1”

Control: radiobutton2
Propiedad: “String”   Nuevo Valor: “100 al 200”

Control: radiobutton3

Propiedad: “String”   Nuevo Valor: “5 en 5 al 50”

Ordene los controles para que luzcan de la siguiente manera.



28.- Es necesario considerar que al utilizar los radio botton en MatLab uno debe actualizar en código el estado de esto, esto significa que al momento de dispararse el evento callback de un radio button, el programador debe actualizar los valores de todos los otros que conciernen al radio button presionado.

Por lo tanto será necesario agregar el siguiente código a las funciones callback del los tres radio botton.

Control: radiobutton1
Función: Callback

```
serie = 1:1:10;
set(handles.listbox1,'string',serie)
set(handles.radiobutton1,'value',1)
set(handles.radiobutton2,'value',0)
set(handles.radiobutton3,'value',0)
```

Control: radiobutton2
Función: Callback

```
serie = 100:1:200;
set(handles.listbox1,'string',serie)
set(handles.radiobutton1,'value',0)
set(handles.radiobutton2,'value',1)
set(handles.radiobutton3,'value',0)
```

Control: radiobutton3
Función: Callback

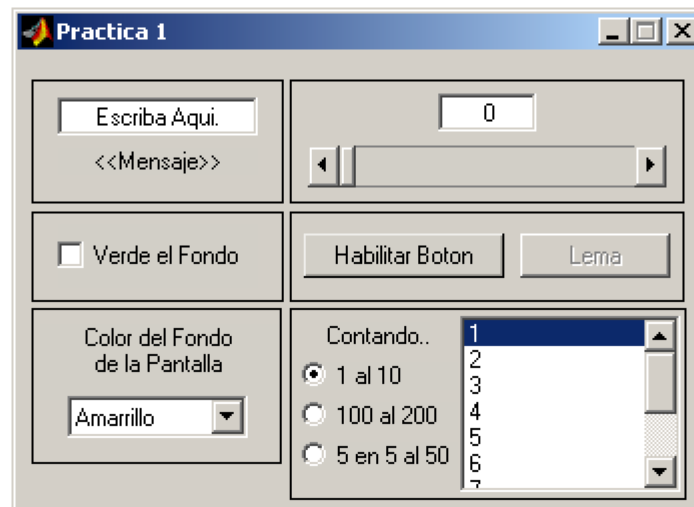
```
serie = 0:5:50;
set(handles.listbox1,'string',serie)
set(handles.radiobutton1,'value',0)
set(handles.radiobutton2,'value',0)
set(handles.radiobutton3,'value',1)
```

29.- Ejecute la figura. Y verifique que funcione.

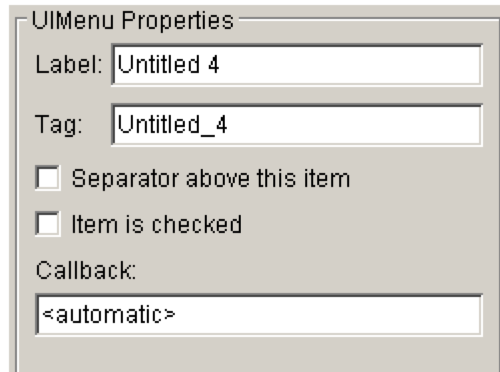
30.- Notara que cuando empieza la GUI no tiene datos en el listbox, aunque tiene seleccionado por default el radio button 1. Para dar este efecto será necesario agregar código a la función de llamado CreateFcn del ListBox de esta manera lo inicializaremos.

```
serie = 1:1:10;
set(h,'string',serie)
```

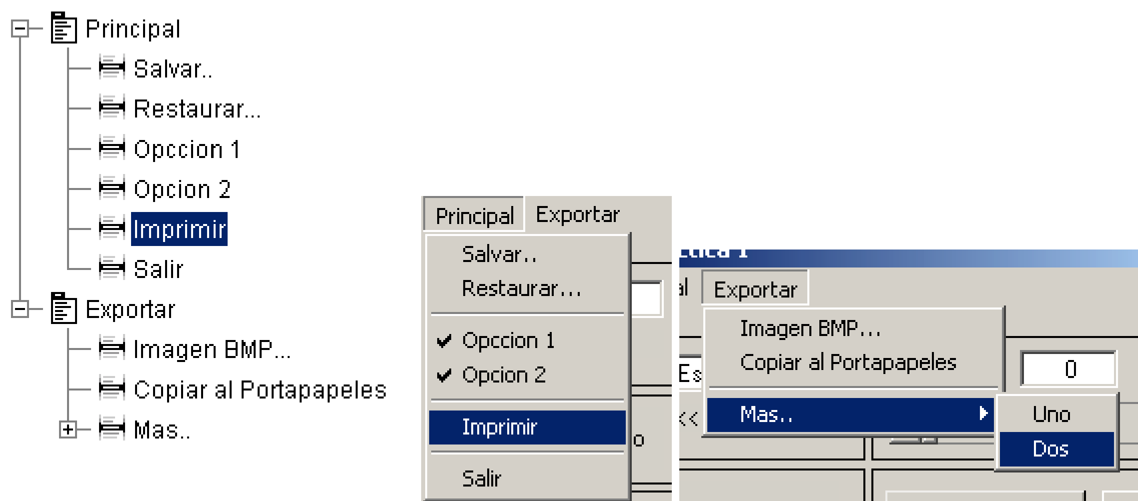
30.- Ejecute la figura. Y en esta ocasión si tendrá datos iniciales



31.- Implementaremos menús para la interfaz a través de Menú Editor . Llamémoslo presionando el icono en la barra de herramientas. Una vez abierto crearemos un menú llamada Archivo, los menús principales se crean presionando el boton y los submenús con . Con las flechas movemos la posición de los menús . Las propiedades de los UIMenu son Label que es el mensaje a mostrar, Tag la etiqueta de identificación, el primer check box coloca un separador en la parte superior de menú, el segundo marca el menú con una palomita, y el edit box de Callback va el nombre de la función.



Creer el siguiente árbol de menús con su respectivo efecto



32.- Menú Editor no da soporte creando la función Callback en el archivo-M, por lo tanto tendremos que crearlo por nosotros mismos.
Cuando Cerramos y Volvemos a abrir el Menú Editor el pondrá automáticamente el nombre de las función en el edit box de callback ya que las dejamos con su valor por default <automatic>. Busque el nombre del control de UIMENU referente al menú de Salir. Y escriba el siguiente código:

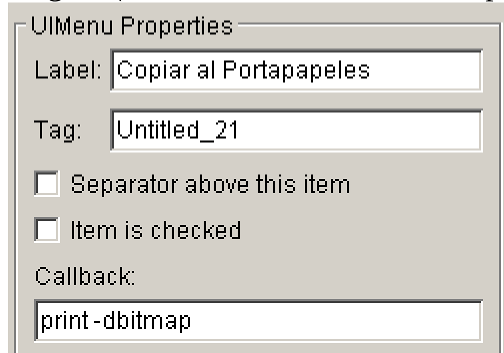
delete(gcf)

33.- Compruebe que se cierra la figura.

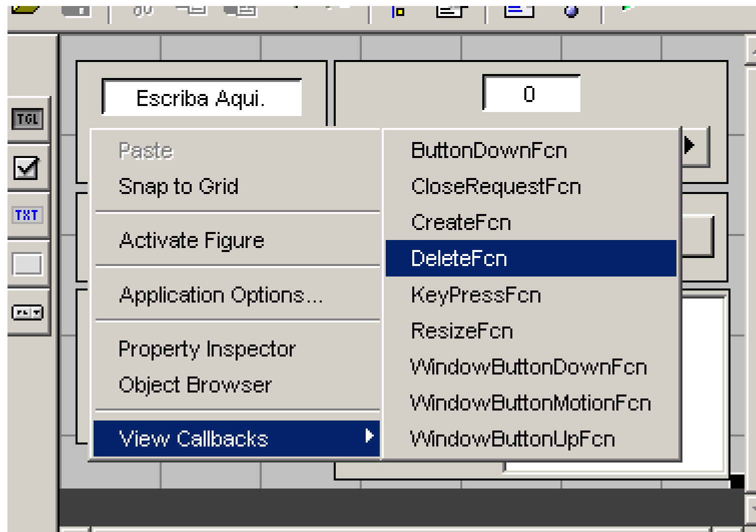
34.- Agregue las siguientes instrucciones de una línea en los Callback's de los menús correspondientes:

Menú: Copiar al portapapeles
Callback: print –dbitmap

Menú: Archivo TIF…
Callback: print temp.tif ; msgbox('Se Gravo en el Archivo temp.tif')

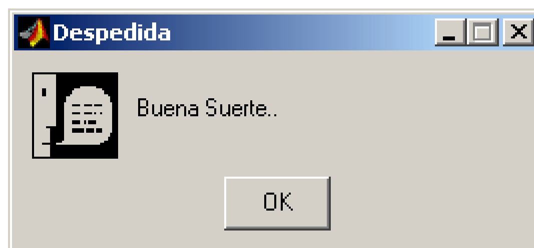


31.- Por Ultimo Agregaremos un mensaje de despedida, que será desplegado al momento de cerrar la figura. Esto se lograra a través de la función de llamado DeleteFcn del objeto figura, se llega a el dando clic derecho sobre el área de trabajo.



En esa función escriba:
msgbox('Buena Suerte..','Despedida','help')

26.- Ejecute la figura. Y Compruebe el Mensaje de Despedida.



Practica 2. “Usando los ejes”.

1.- Primero Insertaremos todos los controles que utilizaremos: 1 frame, 1 static text, 2 radio botton, 2 axes, 3 push botton. Modifique las propiedades de los controles con los siguientes datos.

Control: radiobutton1
Propiedad: “String” Nuevo Valor: “Plano Izquierdo”
Propiedad: “Value” Nuevo Valor: “1”
Propiedad: “Position → Width” Nuevo Valor: “21”.

Control: radiobutton2
Propiedad: “String” Nuevo Valor: “Plano Derecho”
Propiedad: “Position → Width” Nuevo Valor: “21”.

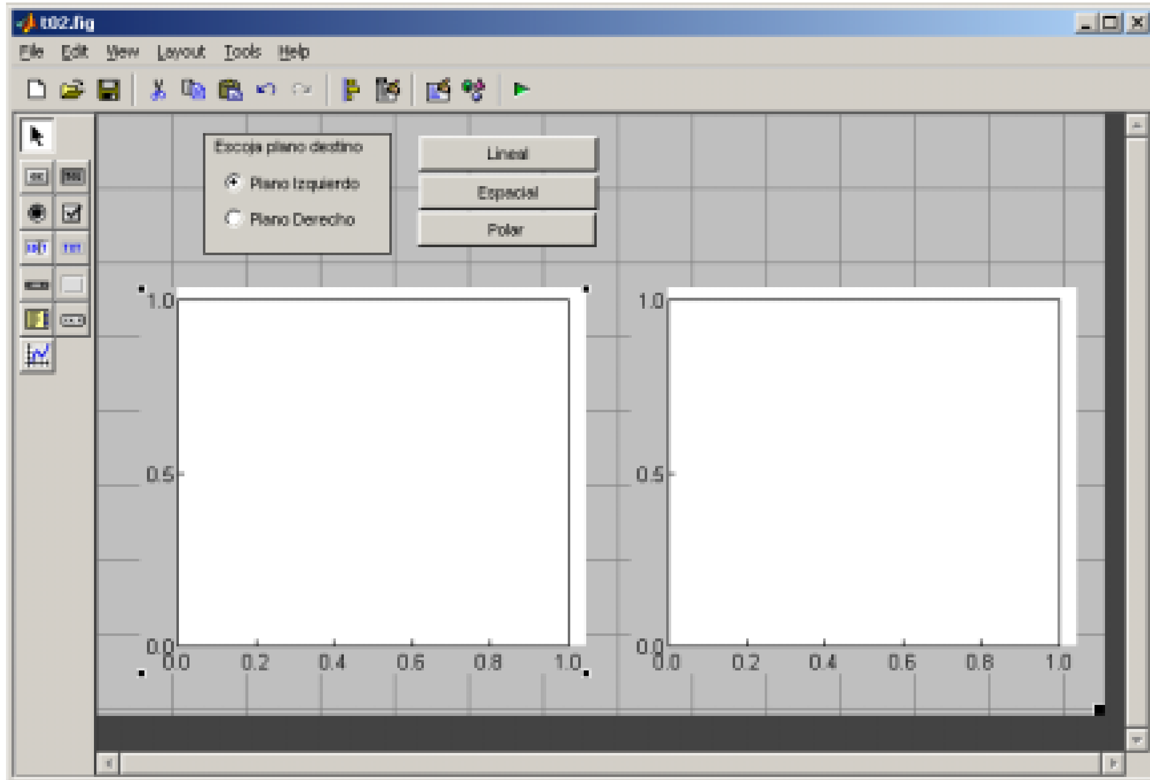
Control: text1
Propiedad: “String” Nuevo Valor: “Escoja plano destino”
Propiedad: “Position → Width” Nuevo Valor: “22”.

Control: pushbutton1
Propiedad: “String” Nuevo Valor: “Lineal”
Propiedad: “Position → Width” Nuevo Valor: “24”.

Control: pushbutton2
Propiedad: “String” Nuevo Valor: “Espacial”
Propiedad: “Position → Width” Nuevo Valor: “24”.

Control: pushbutton1
Propiedad: “String” Nuevo Valor: “Polar”
Propiedad: “Position → Width” Nuevo Valor: “24”.

Ordene los controles como se muestra



2.- Escriba el siguiente código en los callback de los controles:

Control: radiobutton1
```
axes(handles.axes1)  %con esto selecciona el eje a utilizar.
set(h,'value',1);
set(handles.radiobutton2,'value',0);
```

Control: radiobutton2
```
axes(handles.axes2)
set(h,'value',1);
set(handles.radiobutton1,'value',0);
```

Control: pushbutton1
```
newplot % Con esta line reinicializa el axes
x = 0:pi/100:2*pi;
y = sin(x);
y2 = sin(x-.25);
y3 = sin(x-.5);
plot(x,y,x,y2,x,y3)
legend('sin(x)','sin(x-.25)','sin(x-.5)')
xlabel('x = 0:2\pi')
ylabel('Seno de x')
```

```
title('Graficando Funciones Senoidales','FontSize',12)
```

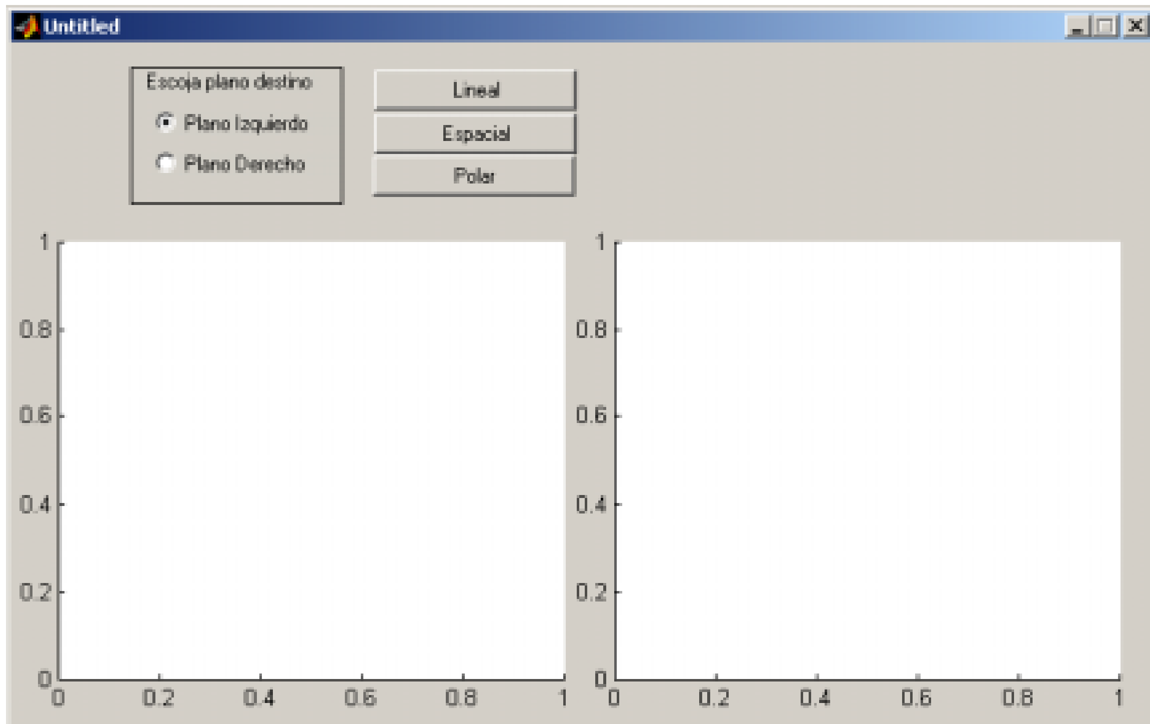
Control: pushbutton2

```
newplot
t = 0:pi/50:10*pi;
plot3(sin(t),cos(t),t)
axis square; grid on
```

Control: pushbutton3

```
newplot
t = 0:.01:2*pi;
polar(t,sin(2*t).*cos(2*t),'--r')
```

3.- Ejecute y pruebe la GUI.



Nota: Los axes se manipulan de igual manera que en los archivos-m, solo indique que eje es el destino de sus comandos antes.

APENDICE.

UICONTROLS

BackgroundColor

El color usado para rellenar el rectángulo de unicontrol. Específica un color usando un vector de tres elementos RGB(rojo, verde y azul) o uno de los nombres ya predefinidos en Matlab. El color por "default" es determinado por la configuración del sistema

BusyAction

Interrupción de la rutina de llamada(callback). Si una es ejecutada y el usuario activa un evento en un objeto para el cual una llamada esta definida, esa llamada trata de interrumpir la primera llamada. La primera llamada puede ser interrumpida solamente por uno de los siguientes comandos: drawnow, figure, getframe, pause o waitfor; si la llamado no contiene ninguno de estos comandos no puede ser interrumpida.

Si la propiedad Interruptible de el objeto que se esta ejecutando la llamada esta desactivada (off), la llamada no puede ser interrumpida (exepto por algunas llamadas).
La propiedad *BusyAction* del objeto que su llamada esta esperando para ejecutarse determina lo que le pasa a la llamada:

Si el valor es queue, la llamada es agregada al evento queue y se ejecuta después de que la primera llamada termina de ejecutarse.

1 Si el valor es *Cancel*, el evento es descartado y la llamada no se ejecuta.

<u>Nota</u>: Si la llamada interrumpida es una llamada de *DeleteFcn* o *CreateFcn* o una de una figura de *CloseRequest* or *ResizeFcn*, se interrumpe y ejecuta sin importar el valor de la propiedad Interrumpible del objeto

ButtonDownFcn

Una rutina de llamada que se ejecuta cuando presionas un botón del mouse mientras el cursor esta en un unicontrol. Cuando la propiedad enable del unicontrol esta desactivada, el *ButtonDownFcn* se ejecuta cuando haces click en el unicontrol. Esto es útil para implementar acciones para modificar interactivamente las propiedades de control del objeto, como el tamaño y la posición.

Esta rutina se define como una cadena(string) que es una expresión valida en Matlab o el nombre de un archivo M (M-file). La expresión se ejecuta en el espacio de trabajo de Matlab.

La propiedad de llamada define la rutina de llamada que se ejecuta cuando das click en el botón

Callback
Controla la acción. Una rutina que se ejecuta cuando se activa un objeto de la clase uicontrol.
Define esta rutina como una cadena. La expresión se ejecuta en el espacio de trabajo de Matlab.

Para ejecutarla rutina para un control de texto editable, escribe el texto deseado y después sigue uno de los siguientes pasos:

-Mueve la selección del objeto (da click en cualquier otra parte)
-Para un texto editable de una sola línea, presiona Return
-Para una caja de texto (text box), presiona Ctrl-Return.

Esta rutina definida para los componentes frame y ststic text no se ejecuta por que ninguna acción esta asociada con estos objetos.

Cdata
Imagen de color verdadero mostrada en un control. Una matriz tridimensional de valores RGB que definen una imagen de color verdadero que es mostrada ya sea en un push button o un toggle button. Cada valor debe tener un rango entre cero y uno.

CreateFcn
Rutina de llamada ejecutada cuando se crea un objeto. Esta propiedad define una rutina de llamada que es ejecutada cuando Matlab crea un objeto de la clase uicontrol. Se debe definir esta propiedad como un valor por default para los uicontrols.

DeleteFcn
Una rutina de llamada que se ejecuta cuando borras un objeto uicontrol. Matlab ejecuta la rutina antes de destruir las propiedades del objeto, así sus valores están disponibles para la rutina de llamada.

Enable
Activa o desactiva el uicontrol. Esta propiedad controla como los uicontrols responden a un click del mouse, incluyendo que rutina de llamada se ejecuta.

on - El uicontrol es oreracional
inactive - no es operacional pero se ve como si estubiera activado
off - No es operacional y su etiqueta se vuelve gris

Cuando se da click izquierdo a un objeto uicontrol que tiene su propiedad enable activada (en on), Matlab ejecuta estas en este orden:

- Asigna la propiedad de la figura *SelectionType*
- Ejecuta la rutina de llamada de el control.
- No asigna la propiedad de la figura *CurrentPoint* y tampoco ejecuta ni la propiedad de control *ButtonDownFcn* ni la rutina de llamada de la figura *WindowButtonDownFcn*

Cuando se da click izquierdo en un uicontrol en el cual su propiedad Enable esta inactiva, o cuando das click derecho en uno en el que Enable tiene cualquier valor, Matlab ejecuta estas acciones in este orden:

- Asigna la propiedad de la figura *SelectionType*.
- Asigna la propiedad de la figura *CurrentPoint*.
- Ejecuta la rutina de llamada *WindowButtonDownFcn* de la figura.
- En un click derecho, si el uicontrol esta asociado con un context menu, registra el context menu
- Ejecuta la llamada *ButtonDownFcn* del control
- Ejecuta la llamada del elemento seleccionado del context menu
- No ejecuta la rutina de llamada del control

Poniendo esta propiedad inactiva te capacita para implementar arrastre o cambio de tamaño de objetos usando la rutina de llamada *ButtonDownFcn*

Extent

Tamaño de un caracter cadena uicontrol. Un vector de cuatro elementos que define el tamaño y la posicion de un caracter de tipo cadena usado para etiquetar el uicontrol. Tiene la forma:

[0, 0, width,height]

Los dos primeros elementos siempre son cero. width (ancho) y height (alto) son las dimenciones del rectángulo. Todas las medidas son unidades especificadas por la propiedad *Units.*

Ya que la propiedad *Extent* esta definida en las mismas unidades que el uicontrol mismo, puedes usar esta propiedad para determinar el tamaño adecuado del ancho del uicontrol con respecto a su etiqueta. Haciendo lo siguiente:

- Definiendo la propiedad *String* y seleccionando la fuente usando las propiedades relevantes.
- Tomando el valor de la propiedad *Extend*
- Definiendo *width* y *height* de la propiedad *Position* (posicion) propiamente a ser de alguna manera mas grandes que *width* y *height de Extend.*

FontAngle
Inclinacion de un Caracter. Poniendo esta propiedad en *Italic* (italica) o *oblique* (oblicua) selecciona una version inclinada de la fuente, cuando esta disponible en tu sistema.

FontName
El nombre de la fuente que mostrara la cadena. Para mostrar e imprimir correctamente, debe ser un tipo de fuente que tu sistema soporte.
Para usar un ancho ajustado que se vea bien en cualquier exterior (y que se muestre correctamente en Japon, donde usan caracteres "multibyte"), Pon al FontName la cadena FixedWidth (esta cadena es sencible a la mayusculas):
Set (uicontrol_handle, 'FontName', 'FixedWidth')

FontSize
Tamaño de la fuente. Un numero que especifica el tamaño de la fuente que va a ser mostrado en la cadena, in unidades determinadas por la propiedad *FontUnits*. El tamaño por default es dependiente del sistema.

FontUnits
Unidades del tamaño de la fuente. Esta propiedad determina las unidades usadas por la propiedad *FontSize*.Las unidades normalizadas interpretan el *FontSize* como una fraccion de la altura de el uicontrol. Cuando tu cambias el tamaño del uicontrol, Matlab modifica la pantalla *FontSize*. *pixels* (pixeles), *inches* (pulgadas), *centimeters* (centimetros) y *points* (puntos) son unidades absolutas (1 punto = $^{1}/_{72}$ pulgada).

FontWeight
Peso de un caracter. Poniendo esta propiedad en Bold hace que Matlab use una version "negrita" de la fuente, cuando esta disponible en tu sistema.

ForegroundColor
Color de texto. Esta propiedad determina el color de el texto definido por la propiedad String. Especifica un color usando un vector de tres elementos RGB o un nombre predefinido en Matlab.

HandleVisibility
Controla el acceso al manejador (handle) de un objeto por usuarios de la linea de comando y GUI's. Esta propiedad determina cuando un manejador de un objeto es visible

en la lista de los objetos hijos de su clase papa.

HandleVisibility es util para prevenir usuarios de la linea de comando de accidentalmente borrar o dibujar en una figura que contiene solo dispositivos de interfase de usuarios.

Los identificadores son siempre visibles cuando HandleVisibilty esta activada (en on).
Asignando HandleVisibility a una llamada hace que el manejador sea visible para rutinas de llamada o funciones invocadas po rutinas de llamada, pero no para las que son invocadas desde la linea de comando. Esto es para proteger los GUI's de los usuarios de la linea de comando, y tambien permite a las rutinas de llamada tener el completo acceso a los identificadores de los objetos.

Poniendo HandleVisibility en off hace al manejador siempre invisible. Esto puede ser necesario cuando una rutina de llamada invoca a una funcion que puede dañar al GUI, por que asi temporalmente esconde sus propios identificadores mientras se ejecuta dicha funcion.
Cuando un manejador no es visible en la lista de hijos de su clase papa, no puede ser regresado por funciones que obtengan identificadores buscando la jerarquia del objeto o "preguntando" las propiedades del manejador. Esto incluye *get, findobj, gca, gcf, gco, newplot*, *cla, clf* y *close.*

Cuando la visibilidad del manejador (HandleVisibility) es restringida usando llamadas o poniendola en off, el manejador del objeto no aparece en la propiedad children de su papa, las figuras no aparecen en la propiedad *CurrentFigure* de *Root* (raiz), los objetos no aparecen en la propiedad de *Root CallbackObjet* o en la propiedad de la figura *CurrentObjet*, y los *Axes* (ejes) no aparecen en la propiedad *CurrentAxes* de sus padres.

Puedes poner la propiedad de Root *ShowHiddenHandles* en on para hacer visibles a todos los identificadores, sin importar los ajustes de su propiedad *HandleVisibility* (esto no afecta sus valores).

Los identificadores que estan escondidos siguen siendo validos, si se conoce el manejador de un objeto tu puedes asigner (set) y obtener (get) sus propiedades, y pasarcelas a cualquier funcion que opere identificadores

HorizontalAlignment
Alineacion Horisontal de una cadena de una etiqueta. Esta propiedad determina la justificaion de el texto definido por la propiedad *String* (la etiqueta uicontrol):

- *left* - El texto se justifica a la izquierda con respecto al uicontrol
- *center* - El texto se centra con respecto al uicontrol
- *right* - El texto se justifica a la derecha con respecto al uicontrol

En el sistema operativo Windows de Microsoft, esta propiedad afecta solo a los uicontrols edit y text.

Interruptible

Rutina de llamada modo de interupcion. Si una llamada se esta ejecutando y el usuario dispara un evento (como un click del mouse) en un objeto para el cual esta definida uina llamada, la llamada intenta interumpir la primera llamada. Matlab procesa las llamadas conforme a estos factores:

La propiedad Interruptible de el objeto que se esta ejecutando la llamada

- Si la llamada que se esta ejecutando contiene las declaraciones *drawnow, figure, getframe, pause o waitfor*
- LA propiedad *BusyAction* de el objeto que su llamada esta esperando para ejecutarse

Si la propiedad *Interruptible* de el objeto que su llamada se esta ejecutando esta en on, la llamada puede ser interrumpida. La llamada interumpe ejecucion a la siguiente declaracion *drawnow, figure, getframe, pause o waitfor*, y procesa los eventos en el evento de la cola, en el cual incluye la llamada que esta espreando.

Si la propiedad Interruptible de el objeto que su llamada se esta ejecutando esta en off, la llamada no puede ser interrumpida (a excepcio de ciertas llamadas). La propiedad *BusyAction* de el objeto de el cual su llamada esta esperando para ejecutarse determina que pasa con la llamada.

Nota: Si la llamada interumpida es una llamada de *DeleteFcn* o *CreateFcn* o una llamada de *CloseRequest* o *ResizeFcn* de una figura, interrumpe la llamada que se esta ajecutando sin importar el valor de la propiedad Interruptible de ese objeto. La llamada interruptora empiesa la ejecucion a la siguiente declaracion de *drawnow, figure, getframe, pause o waitfor*. la rutina de llamada *WindowButtonDownFcn* de una figura, o las propiedades *ButtonDownFcn* o *Callback* de un objeto son procesadas de acuerdo a las reglas descritas arriba.

ListboxTop

Esta propiedad se aplica solo al estilo de uicontrol listbox. Especifica que cadena aparece en la posicion mas alta de un Listbox (caja de lista) que no es lo suficientemente largo para mostrar todos los elementos de la lista. ListboxTop es un indice en el arreglo de cadenas definido por la propiedad String y debe tener un valor entre cero y el numero de cadenas. Valores no enteros son redondeados al proximo valor entero mas chico.

Max

Valor Maximo. Esta propiedad especifica el valor mas grande aceptado por la propiedad Value.

Diferentes estilos de uicontrols interpretan la propiedad max de diferente manera:

- *Check boxes* - Max es el valor de la propiedad Value mientras el check box esta seleccionado.
- *Editable text* - Si Max - Min > 1, entonces el editable text (texto editable) ecepta varias lineas de entrada. Si Max - Min <= 1, entonces el editable text solo aceptan una linea de entrada.
- *List boxes* - Si Max - Min > 1, entonces el list box acepta seleccion multiple de elementos. Si Max - Min <= 1, entonces no aceptan seleccion multiple
- *Radio buttons* - Max es el valor de la propiedad Value mientras el radio button esta seleccionado.
- *Sliders* - Max es el valor maximo del slider (deslisador) y tiene que ser mas grande que la propiedad min. El valor por defecto es 1.
- *Toggle buttons* - Max es el valor de la propiedad Value mientras el radio button esta seleccionado. El valor por defecto es 1.
- *Frames, pop-up menus, push buttons*, y *static text* no usan la propiedad Max

Min

Valor Maximo. Esta propiedad especifica el valor mas pequeño aceptado por la propiedad *Value*.

Diferentes estilos de uicontrols interpretan la propiedad *min* de diferente manera:

- *Check boxes* - Min es el valor de la propiedad *Value* mientras el check box no esta seleccionado.
- *Editable text* - Si Max - Min > 1, entonces el *editable text* (texto editable) ecepta varias lineas de entrada. Si Max - Min <= 1, entonces el *editable text* solo aceptan una linea de entrada.
- *List boxes* - Si Max - Min > 1, entonces el *list box* acepta seleccion multiple de elementos. Si Max - Min <= 1, entonces no aceptan seleccion multiple
- *Radio buttons* - Min es el valor de la propiedad *Value* mientras el *radio button* no esta seleccionado.
- *Sliders* - Min es el valor minimo del *slider* (deslisador) y tiene que ser menos que la propiedad *max*. El valor por defecto es 0.
- *Toggle buttons* - Min es el valor de la propiedad *Value* mientras el *radio button* no esta seleccionado. El valor por defecto es 0.
- *Frames, pop-up menus, push buttons*, y *static text* no usan la propiedad Min

Parent

Papa de un unicontrol. El manejador de el objeto padre del uicontrol. El papa de un objeto uicontrol es la figura (figure) en el que aparece. Puedes mover un objeto uicontrol a otra figura asignando esta propiedad el manejador de el nuevo padre.

Position

Tamaño y posicion de un uicontrol. El rectangulo definido por esta propiedad especifica el tamaño y la posicion de el control en la ventana de la figura. Especifica la posicion

como:

[left bottom width height]

left (izquierda) y bottom (abajo) son la distancia de la esquina izquierda de abajo de la figura a la esquina izquierda de abajo del objeto. width (ancho) y height (alto) son las dimenciones de el rectangulo uicontrol. Todas las medidas estan en unidades especificadas por la propiedad Units.

En el sistema operativo Windows de Microsoft, la altura de los pop-up menus es automaticamente determinada por el tamaño de la fuente. El valor que tu especifiques para height de la propiedad Position no tiene ningun efecto.

Los valores de width y height determinan la orientacion de los sliders (deslisadores). Si width es
mas grande que height, entonces el slider es orientado horizontalmente, de lo contratio es orientado de manera vertical.

Selected

Objeto seleccionado. Cuando esta propiedad esta activada (en on), Matlab muestra identificadores de seleccion si la propiedad *SelectionHighlight* esta tambien en on. Tu puedes, por ejemplo, definir *ButtonDownFcn* para que ajuste a esta propiedad, permitiendo al usuario que seleccione el objeto con el mouse.

SelectionHighlight

Objeto resaltado cuando esta seleccionado. Cuando la Propiedad esta en on, Matlab indica el estado de seleccionado dibujando cuatro identificadores de filo y cuatro de horilla. Cuando la propiedad esta en off, Matlab no dibuja los identificadores. ****

SliderStep

Paso del slider (deslizador). Esta propiedad controla la cantidad que la propiedad del slider *Value* cambia cuando le das click en el boton de la flecha (min_step) o en la barra del slider (max_step). Especifica *SliderStep* como un vector de dos elementos; cada valor debe estar en un rango de cero y uno. El tamaño del paso es una funcion de el especificado *SliderStep* y al rango total del slider (Max - Min). El valor por defecto, [0 .01 0 .10], proporciona 1 por ciento oportunidades de clicks en el boton de la flecha y 10 porciento en la barra.

Por ejemplo, si creas el siguiente slider,

```
uicontrol ('style', 'slider', 'min', 1, 'max', 7,... 'SliderStep', [0 .1 0 .6] )
```

dando click en el boton de la flecha mueve el indicador por,

```
0.1*(7-1)
ans =
        0.6
```

y dando click en la barra mueve el indicador

```
0.6*(7-1)
ans =
        3.6
```

Nota: que si el tamaño del paso especificado mueve el slider a un valor fuera del rango, el indicador se muve solo al valor max o min.

String

Para los check boxes, editable text, push buttons, radio buttons, static text, and toggle buttons, el texto que se muestra en el objeto. Para los list boxes y pop-up menu, el conjunto de elementos o articulos del objeto.

Para objetos uicontrol que muestran solo una linea de texto, si el valor de String es especificado como arreglo de celdas tipo cadena (los elementos en las celdas son cadenas) o una matriz rellena, solo la primer cadena se muestra; los demas son ignorados, el caracter guion vertical ('|') no es interpretado como un cambio de linea sino que se muestra en el texto del uicontrol. Para texto editable o estatico(editable text o static text), cambios de linea ocurren entre cada columna de la matriz, cada celda de el arreglo tipo cadena y despues de cada caracter \n.

Para multiples elementos en un list box o un pop-up menu, tu puedes especificar los elementos como un arreglo de celdas tipo cadenas, una matriz rellena de cadenas, o en un vector separados por guiones verticales ('|').

Para editable text, el valor de esta propiedad es asignado a la cadena capturada por el usuario.

Style

Estilo del objeto uicontrol a crear. La propiedad Style (estilo) especifica el tipo de uicontrol a crear.

Tag

Etiqueta del objeto especificada por el usuario. La propiedad *tag* proporciona una manera de identificar objetos graficos con una etiqueta espesificada por el usuario. Esta es particularmente util cuando construyes programas graficos interactivos que de otra manera necesitarian definir identificadores de objetos como variables globales o pasarlas como argumentos entre rutinas de llamada. Puedes definir la propiedad tag como cualquier cadena.

TooltipString

Contenido de la "pista" (el letrero que aparece si dejas el mouse sobre un objeto sin dar click). La propiedad *TooltipString* especifica el texto de la "pista" asociado con el uicontrol. Cuando el usuario mueve el puntero mouse sobre un control y lo deja hay, la pista aparece.

Type

Clase de objeto grafico. Para los objetos uicontrol, la propiedad *Type* siempre es la cadena 'uicontrol'.

UIContextMenu
Asocia un context menu con uicontrol. Asigna a esta propiedad el manejador de un objeto *uicontextmenu*. Matlab muestra el context menu cuando das click derecho en el uicontrol. Usa la funcion *uicontextmenu* para crear el context menu.

Units
Unidades de medida. Las unidades que Matlab usa para interpretar las propiedades *Extend* y *Position.* Todas las unidades son medidas de la esquina inferior izquierda de la figura ventana.

UserData
Datos especificados por el usuario. Cualquier deto que quieras asociar con un objeto uicontrol. Matlab no usa este dato pero tu puedes accesar a el utilisando los comandos *set* (asigna) y *get* (recupera).

Value
Valor actual de el uicontrol.La clase (o estilo) uicontrol determina los posibles valores que esta propiedad puede tener:

- *Check boxes* - ponen su propiedad *Value* (valor) en *Max* (el maximo) cuando estan seleccionados y en *Min* (el minimo) cuando no estan selaccionados.
- *List boxes* - ajustan su valor a un vector corrspondiente a los elementos de la lista
- *Pop-up menu* - ajustan su valor a un indice de articulos seleccionados , donde 1 corresponde al primer elemento del menu.
- *Radio buttons* - ponen su propiedad *Value* (valor) en *Max* (el maximo) cuando estan seleccionados y en *Min* (el minimo) cuando no estan selaccionados.
- *Sliders* - ajustan su valor al numero indicado por la posicion de el *slider* (deslisador)
- *Toggle buttons* - ponen su propiedad *Value* en *Max* cuando estan presionados (seleccionados) y en *Min* cuando no estan selaccionados.
- *Editable text, Frames, Push buttons, y Static text* no usan esta propiedad.

Puedes ajustar este valor ya sea interactivamente con el mouse o atraves de la funcion set.

Visible
Visibilidad de el uicontrol. Por defecto, todos los uicontrols son visibles. Cuando esta propiedad
propiedad esta desactivada (en *off*), el control no es visible, pero sigue existiendo y puedas buscar y modificar su propiedades.

UIMENUS

Accelerator

Un caracter que espesifica la tecla equivalenteal articulo del menu. Esto permite a los usuarios seleccionar un elemento en particular persionando un caracter especifico en conjuncion con otra tecla, en lugar de hacerlo con el mouse. La secuencia de las teclas depende de la plataforma:

- Para Windows de Microsoft, la secuencia es Ctrl-Accelerator. Las teclas reservadas para los elementos del menu por defecto son: *c, v,* y *x*
- Para Unix, la secuencia es Ctrl-Accelerator. Las teclas reservadas para los elementos del menu por defecto son: *o, p, s,* y *w*

Puedes definir un accelerator (acelerador) solamente para elementos que no tienen menus hijos. Los Accelerators solo funcionan para elementos del menu que ejecutan una rutina de llamada directamente, y no para los que abren otros menus.
Note que los elementos de el menu no tienen que estar a la vista para que el accelerator funcione. Pero la figura papa de el menu tiene que estar seleccionada.

BusyAction

Interrupcion de la rutina de llamada(callback). Si una es ejecutada y el usuario activa un evento en un objeto para el cual una llamada esta definida, esa llamada trata de interumpir la primera llamada. La primera llamada puede ser interrumpida sola mente por uno de los siguientes comandos: drawnow, figure, getframe, pause o waitfor; si la llamado no contiene ninguno de estos comandos no puede ser interrumpida.

Si la propiedad Interruptible de el objeto que se esta ejecutando la llamada esta desactivada(off), la llamada no puede ser interrumpida (exepto por algunas llamadas).
La propiedad *BusyAction* de el objeto que su llamada esta esperando para ejecutarse determina lo que le pasa a la llamada:

Si el valor es queue, la llamada es agregada al evento queue y se ejecuta después de que la primera llamada termina de ejecutarse.

1 Si el valor es *Cancel,* el evento es descartado y la llamada no se ejecuta.

Nota: Si la llamada interrumpida es una llamada de *DeleteFcn* o *CreateFcn* o una de una figura de *CloseRequest* or *ResizeFcn,* se interrumpe y ejecuta sin importar el valor de la propiedad Interruptible del objeto

ButtonDownFcn
Rutia de llamada de presionar un boton. Una rutina de llamada que se ejecuta cuando presionas un boton del mouse mientras el cursor esta en un unicontrol. Cuando la propiedad enable del unicontrol esta desactivada, el *ButtonDownFcn* se ejecuta cuando haces click en el unicontrol. Esto es util para implementar acciones para modificar interactivamente las propiedades de control del objeto, como el tamaño y la posición.

Esta rutina se define como una cadena(string) que es una expresión valida en Matlab o el nombre de un archivo M (M-file). La expresión se ejecuta en el espacio de trabajo de matlab.

La propiedad de llamada define la rutina de llamada que se ejecuta cuando das click en el boton

Callback
Controla la accion. Una rutina que se ejecuta cuando se activa un objeto de la clase uicontrol.
Define esta rutina como una cadena. La exprecion se ejecuta en el espacio de trabajo de matlab.

Para ejecutarla rutina para un control de texto editable, escribe el texto deseado y despues suige uno de los siguientes pasos:

-Mueve la seleccion del objeto (da click en cualquier otra parte)
-Para un texto editable de una sola linea, presiona Return
-Para una caja de texto (text box), presiona Ctrl-Return.

Esta rutina definida para los componentes frame y ststic text no se ejecuta por que ninguna accion esta asociada con estos objetos.

Checked
Indicador del articulo del menu seleccionado. Poniendo esta propiedad en *on* muestra una marca a un lado de el elemanto de el menu correspondiente. Poniendola en *off* remueve la marca. Tu puedes usar esta opcion para crear menus que indiquen el estado de una opcion en particular.

Children
Manejador de submenus. Un vector que contiene los identificadores de todos los hijos del objeto uimenu. Los objetos hijos de los uimenus son otros uimenus, que funcionan como submenus.
Esta propiedad es util para reordenar los menus

CreateFcn
Rutina de llamada ejecutada cuando se crea un objeto. Esta propiedad define una rutina de llamada que es ejecutada cuando matlab crea un objeto de la clase uimenu. Se debe definir esta propiedad como un valor por default para los uicontrols.

DeleteFcn
Una rutina de llamada que se ejecuta cuando borras un objeto uicontrol. Matlab ejecuta la rutina antes de destruir las propiedades del objeto, asi sus valorea estan disponibles para la rutina de llamada.

Enable
Activa o desactiva el uimenu. Esta propiedad controla cuando un elemento del menu puede ser seleccionado. Cuando esta inactiva (en off), la etiqueta del menu aparece osbcurecida, indicandoque no puede ser seleccionada por el usuario

ForegroundColor
Color de texto. Esta propiedad determina el color de el texto definido por la propiedad String. Especifica un color usando un vector de tres elementos RGB o un nombre predefinido en Matlab.

HandleVisibility
Controla el acceso al manejador (handle) de un objeto por usuarios de la linea de comando y GUI's. Esta propiedad determina cuando un manejador de un objeto es visible en la lista de los objetos hijos de su clase papa.

HandleVisibility es util para prevenir usuarios de la linea de comando de accidentalmente borrar o dibujar en una figura que contiene solo dispositivos de interfase de usuarios.

Los identificadores son siempre visibles cuando HandleVisibilty esta activada (en on).
Asignando HandleVisibility a una llamada hace que el manejador sea visible para rutinas de llamada o funciones invocadas po rutinas de llamada, pero no para las que son invocadas desde la linea de comando. Esto es para proteger los GUI's de los usuarios de la linea de comando, y tambien permite a las rutinas de llamada tener el completo acceso a los identificadores de los objetos.

Poniendo HandleVisibility en off hace al manejador siempre invisible. Esto puede ser necesario cuando una rutina de llamada invoca a una funcion que puede dañar al GUI, por que asi temporalmente esconde sus propios identificadores mientras se ejecuta dicha funcion.
Cuando un manejador no es visible en la lista de hijos de su clase papa, no puede ser regresado por funciones que obtebgan identificadores buscando la jerarquia del objeto o

"preguntando" las propiedades del manejador. Esto incluye *get, findobj, gca, gcf, gco, newplot, cla, clf* y *close.*

Cuando la visibilidad del manejador (HandleVisibility) es restringida usando llamadas o poniendola en off, el manejador del objeto no aparece en la propiedad children de su papa, las figuras no aparecen en la propiedad *CurrentFigure* de *Root* (raiz), los objetos no aparecen en la propiedad de *Root CallbackObjet* o en la propiedad de la figura *CurrentObjet*, y los *Axes* (ejes) no aparecen en la propiedad *CurrentAxes* de sus padres.

Puedes poner la propiedad de Root *ShowHiddenHandles* en on para hacer visibles a todos los identificadores, sin importar los ajustes de su propiedad *HandleVisibility* (esto no afecta sus valores).

Los identificadores que estan escondidos siguen siendo validos, si se conoce el manejador de un objeto tu puedes asigner (set) y obtener (get) sus propiedades, y pasarcelas a cualquier funcion que opere identificadores

Interruptible
Rutina de llamada modo de interupcion. Si una llamada se esta ejecutando y el usuario dispara un evento (como un click del mouse) en un objeto para el cual esta definida uina llamada, la llamada intenta interumpir la primera llamada. Matlab procesa las llamadas conforme a estos factores:

- La propiedad Interruptible de el objeto que se esta ejecutando la llamada
- Si la llamada que se esta ejecutando contiene las declaraciones *drawnow, figure, getframe, pause o waitfor*
- LA propiedad *BusyAction* de el objeto que su llamada esta esperando para ejecutarse

Si la propiedad *Interruptible* de el objeto que su llamada se esta ejecutando esta en on, la llamada puede ser interrumpida. La llamada interumpe ejecucion a la siguiente declaracion *drawnow, figure, getframe, pause o waitfor*, y procesa los eventos en el evento de la cola, en el cual incluye la llamada que esta espreando.
Si la propiedad Interruptible de el objeto que su llamada se esta ejecutando esta en off, la llamada no puede ser interrumpida (a excepcio de ciertas llamadas). La propiedad *BusyAction* de el objeto de el cual su llamada esta esperando para ejecutarse determina que pasa con la llamada.

Nota: Si la llamada interumpida es una llamada de *DeleteFcn* o *CreateFcn* o una llamada de *CloseRequest* o *ResizeFcn* de una figura, interrumpe la llamada que se esta ajecutando sin importar el valor de la propiedad Interruptible de ese objeto. La llamada interruptora empiesa la ejecucion a la siguiente declaracion de *drawnow, figure, getframe, pause o waitfor*. la rutina de llamada *WindowButtonDownFcn* de una figura, o las propiedades *ButtonDownFcn* o *Callback* de un objeto son procesadas de acuerdo a las reglas descritas arriba.

Label
Etiqueta del menu. Una cadena especificando el texto en la etiqueta de el articulo de el menu. Puedes especificar un mnemonic usando el caracter "&". Cualquier letra siguiente de el "&" aparece subrrayada y se selecciona el articulo del menu cuando oprimes esa letra mientras el menu es visible. El caracter "&" no se muestra. Para mostrarlo en una etiqueta usa dos "&" en la cadena.
Parent
Papa de el uimenu. El manejador de el objeto papa de un uimenu. El papa de un objeto uimenu es la figura en donde se muestra, o el uimenu de el que es un submenu. Puedes mover un objeto uimenu a otra figura asignando esta propiedad el manejador de el nuevo padre

Position
Posicion relativa del uimenu . El valor de la propiedad *position* (posicion) indica el lugar en una barra de menu o en un menu. Los menus de el nivel de arriba son acomodados de izquierda a derecha en la barra de menu dependiendo de el valor de la propiedad position, donde 1 representa la primera posicion de la izquierda. Los elementos individuales en cualquier menu son acomodados de arriba hacia abajo, con 1 representando la primera posicion de arriba.

Separator
Linea de separacion. Activando esta propiedad (*on*) dibuja una linea divisora sobre el articulo del menu

Tag
Etiqueta del objeto especificada por el usuario. La propiedad *tag* proporciona una manera de identificar objetos graficos con una etiqueta espesificada por el usuario. Esta es particularmente util cuando construyes programas graficos interactivos que de otra manera necesitarian definir identificadores de objetos como variables globales o pasarlas como argumentos entre rutinas de llamada. Puedes definir la propiedad tag como cualquier cadena.

Type
Clase de objeto grafico. Para los objetos uimenu, la propiedad *Type* siempre es la cadena 'uimenu'.

UserData
Datos especificados por el usuario. Cualquier matriz que quieras asociar con un objeto uimenu. Matlab no usa este dato pero tu puedes accesar a el utilisando los comandos *set* (asigna) y *get* (recupera).

Visible
Visibilidad de el uimenu. Por defecto, todos los uimenus son visibles. Cuando esta propiedad
propiedad esta desactivada (en *off*), el control no es visible, pero sigue existiendo y puedas buscar y modificar su propiedades.

UICONTEXTMENU

BusyAction
Interrupcion de la rutina de llamada(callback). Si una es ejecutada y el usuario activa un evento en un objeto para el cual una llamada esta definida, esa llamada trata de interumpir la primera llamada. La primera llamada puede ser interrumpida sola mente por uno de los siguientes comandos: drawnow, figure, getframe, pause o waitfor; si la llamado no contiene ninguno de estos comandos no puede ser interrumpida.

Si la propiedad Interruptible de el objeto que se esta ejecutando la llamada esta desactivada(off), la llamada no puede ser interrumpida (exepto por algunas llamadas).
La propiedad *BusyAction* de el objeto que su llamada esta esperando para ejecutarse determina lo que le pasa a la llamada:

Si el valor es queue, la llamada es agregada al evento queue y se ejecuta después de que la primera llamada termina de ejecutarse.

1. Si el valor es *Cancel*, el evento es descartado y la llamada no se ejecuta.

Nota: Si la llamada interrumpida es una llamada de *DeleteFcn* o *CreateFcn* o una de una figura de *CloseRequest* or *ResizeFcn*, se interrumpe y ejecuta sin importar el valor de la propiedad Interruptible del objeto

Callback
Controla la accion. Una rutina que se ejecuta cuando das click derecho a un objeto de la clase uicontextmenu. La rutina se ejecuta inmediatamente despues de que el context menu es registrada. Define esta rutina como una cadena. La exprecion se ejecuta en el espacio de trabajo de matlab.

Children
Los *uimenus* definidos por el *uicontextmenu*.

CreateFcn
Rutina de llamada ejecutada cuando se crea un objeto. Esta propiedad define una rutina de llamada que es ejecutada cuando matlab crea un objeto de la clase uimenu. Se debe definir esta propiedad como un valor por default para los uicontrols.

DeleteFcn
Una rutina de llamada que se ejecuta cuando borras un objeto uicontrol. Matlab ejecuta la rutina antes de destruir las propiedades del objeto, asi sus valorea estan disponibles para la rutina de llamada.

HandleVisibility
Controla el acceso al manejador (handle) de un objeto por usuarios de la linea de comando y GUI's. Esta propiedad determina cuando un manejador de un objeto es visible en la lista de los objetos hijos de su clase papa.

HandleVisibility es util para prevenir usuarios de la linea de comando de accidentalmente borrar o dibujar en una figura que contiene solo dispositivos de interfase de usuarios.

Los identificadores son siempre visibles cuando HandleVisibilty esta activada (en on).
Asignando HandleVisibility a una llamada hace que el manejador sea visible para rutinas de llamada o funciones invocadas po rutinas de llamada, pero no para las que son invocadas desde la linea de comando. Esto es para proteger los GUI's de los usuarios de la linea de comando, y tambien permite a las rutinas de llamada tener el completo acceso a los identificadores de los objetos.

Poniendo HandleVisibility en off hace al manejador siempre invisible. Esto puede ser necesario cuando una rutina de llamada invoca a una funcion que puede dañar al GUI, por que asi temporalmente esconde sus propios identificadores mientras se ejecuta dicha funcion.
Cuando un manejador no es visible en la lista de hijos de su clase papa, no puede ser regresado por funciones que obtebgan identificadores buscando la jerarquia del objeto o "preguntando" las propiedades del manejador. Esto incluye *get, findobj, gca, gcf, gco, newplot, cla, clf* y *close.*

Cuando la visibilidad del manejador (HandleVisibility) es restringida usando llamadas o poniendola en off, el manejador del objeto no aparece en la propiedad children de su papa, las figuras no aparecen en la propiedad *CurrentFigure* de *Root* (raiz), los objetos no aparecen en la propiedad de *Root CallbackObjet* o en la propiedad de la figura *CurrentObjet*, y los *Axes* (ejes) no aparecen en la propiedad *CurrentAxes* de sus padres.

Puedes poner la propiedad de Root *ShowHiddenHandles* en on para hacer visibles a todos los identificadores, sin importar los ajustes de su propiedad *HandleVisibility* (esto no afecta sus valores).

Los identificadores que estan escondidos siguen siendo validos, si se conoce el manejador de un objeto tu puedes asigner (set) y obtener (get) sus propiedades, y pasarcelas a cualquier funcion que opere identificadores

Interruptible
Rutina de llamada modo de interupcion. Si una llamada se esta ejecutando y el usuario dispara un evento (como un click del mouse) en un objeto para el cual esta definida uina llamada, la llamada intenta interumpir la primera llamada. Matlab procesa las llamadas conforme a estos factores:

- La propiedad Interruptible de el objeto que se esta ejecutando la llamada
- Si la llamada que se esta ejecutando contiene las declaraciones *drawnow, figure, getframe, pause o waitfor*
- LA propiedad *BusyAction* de el objeto que su llamada esta esperando para ejecutarse

Si la propiedad *Interruptible* de el objeto que su llamada se esta ejecutando esta en on, la llamada puede ser interrumpida. La llamada interumpe ejecucion a la siguiente declaracion *drawnow, figure, getframe, pause o waitfor,* y procesa los eventos en el evento de la cola, en el cual incluye la llamada que esta espreando.
Si la propiedad Interruptible de el objeto que su llamada se esta ejecutando esta en off, la llamada no puede ser interrumpida (a excepcio de ciertas llamadas). La propiedad *BusyAction* de el objeto de el cual su llamada esta esperando para ejecutarse determina que pasa con la llamada.

Nota: Si la llamada interumpida es una llamada de *DeleteFcn* o *CreateFcn* o una llamada de *CloseRequest* o *ResizeFcn* de una figura, interrumpe la llamada que se esta ajecutando sin importar el valor de la propiedad Interruptible de ese objeto. La llamada interruptora empiesa la ejecucion a la siguiente declaracion de *drawnow, figure, getframe, pause o waitfor*. la rutina de llamada *WindowButtonDownFcn* de una figura, o las propiedades *ButtonDownFcn* o *Callback* de un objeto son procesadas de acuerdo a las reglas descritas arriba.

Parent
Papa de el uicontextmenu. El manejador de el objeto papa de un uicontextmenu. El papa de un objeto uicontextmenu es la figura en donde se muestra. Puedes mover un objeto uicontextmenu a otra figura asignando esta propiedad el manejador de el nuevo padre

Position
Posicion del uicontextmenu . Una vector de dos elementos que define la posicion de el menu.
Espesifica la posicion como: [left bottom]
donde los elementos del vector representan la distancia en pixeles desde la esquina inferior izquierda de la figura hasta la esquina superior izquierda de el context menu.

Tag
Etiqueta del objeto especificada por el usuario. La propiedad *tag* proporciona una manera de identificar objetos graficos con una etiqueta espesificada por el usuario. Esta es particularmente util cuando construyes programas graficos interactivos que de otra

manera necesitarian definir identificadores de objetos como variables globales o pasarlas como argumentos entre rutinas de llamada. Puedes definir la propiedad tag como cualquier cadena.

Type
Clase de objeto grafico. Para los objetos uicontextmenu, la propiedad *Type* siempre es la cadena 'uicontextmenu'.

UserData
Datos especificados por el usuario. Cualquier matriz que quieras asociar con un objeto uicontext menu. Matlab no usa este dato pero tu puedes accesar a el utilisando los comandos *set* (asigna) y *get* (recupera).

Visible
Visibilidad de el uicontextmenu. Esta propiedad puede ser usada de dos maneras:

- El valor indica cuando el contextmenu esta siendo activado.
- El valor puede ser ajustado en *on* para forsar su activacion. De la misma manera puedes poner su valor en *off* para forsar que sea removido. Cuando se usa de esta manera, la propiedad *Position* determina la localidad de el context menu activado

AXES

AmbientLightColor
El color de el fondo en una escena. Es una luz sin direccion que brilla uniformemente en todos los objetos en los ejes (*axes*). Sin embargo, si no hay objetos de luces visibles en los ejes, Matlab no usa *AmbientLightColor*. Si hay objetos de luz en los ejes, el *AmbientLightColor* es agregado a las otras fuentes de luz.

AspectRatio
Esta propiedad produce un aviso de precaucion cuando es buscado o cambiado. A sido remplazada por las propiedades *DataAspectRatio*[Mode] y *PlotBoxAspectRatio*[Mode]

Box
Modo "caja" de los ejes. Esta propiedad especifica si se encierran los ejes en una caja vista en dos dimenciones o en un cubo de tres dimenciones.

BusyAction
Interrupcion de la rutina de llamada(callback). Te permite controlar como Matlab maneja los eventos que potencialmente interumpen ejecutando rutinas de llamada. Si hay una rutina de llamada ejecutandose, las rutinas de llamada invocadas despues siempre tratan de intrrumpirla. Si la propiedad interruptible de el objeto que esta ejecutando la llamada esta activada (en on), entonces la *interrupcion* ocurre en el siguiente punto el evento que esta en la cola es procesado. Si la propiedad interruptible esta en off, la propiedad BusyAction (de el objeto que esta ejecutando la llamada) determina como Matlab maneja el evento. Las opciones son:

- cancel - descarta el evento que trata de ejecutar una segunda rutina de llamada
- queue - pone en una cola al evento que trata de ejecutar la segunda llamada hasta que la llamada actual termine

ButtonDownFcn
Rutia de llamada de presionar un boton. Una rutina de llamada que se ejecuta cuando presionas un boton del mouse mientras el cursor esta en los ejes, pero no sobre otros objetos graficos mostrados en los ejes. Para las vistas en 3-D, el area activa esta definida por un rectangulo que encierra los ejes.

Esta rutina se define como una cadena(string) que es una expresión valida en Matlab o el nombre de un archivo M (M-file). La expresión se ejecuta en el espacio de trabajo de matlab.

CameraPosition
La posicion de la camara. Esta propiedad define la posicion donde la camara ve la escena. Espesifica el punto en coordenadas de los ejes.

CameraPositionMode
Posicion de la camara automatica o manual. Cuando esta an *auto* (automatica), Matlab calcula automaticamente la posicion de la camara. Asignar un valor a *CameraPosition* pone a esta propiedad en *manual*

CameraTarget
Punto de fijacion de la camara. Esta propiedad especifica la posicion en los ejes donde la camara apunta. El *CameraTarget* (blanco de la camara) y la *CameraPosition* (posicion de la camara) define el vector en donde la camara apunta.

CameraTargetMode
Propiedad *CameraTarget* automatica o manual. Cuando Esta propiedad esta en *auto*, Matlab automaticamenta posiciona el lugar donde apunta la camara (*CameraTarget) en el centro de los ejes.* Espesificando un valor a la propiedad *CameraTarget* pone a esta propiedad en *manual*.

CameraUpVector
Rotacion de la camara. Esta propiedad especifica la rotacion de la camara alrededor de los ejes que se estan viendo definida por las propiedades *CameraTarget* y *CameraPosition*. Especifica esta propiedad como un arreglo de tres valores conteniendo los componentes del vector *x, y* y *z.*

CameraUpVectorMode
Vector especificado por el usuario o por defecto. Cuando esta en auto, Matlab usa el valor de
[0 0 1] para vistas en 3-D y [0 0 1] para vistas en 2-D. Asignar un valor a *CameraUpVector* pone esta propiedad en manual.

CameraViewAngle
El campo de vista. esta propiedad Determina el campo de vista de la camara. Cambiando este valor afecta el tamaño de los objetos graficos mostrados en los ejes, pero no afecta el

grado de distorcion de la prespectiva. Entre mas grande sea el angulo, mas grande es el campo de vista, y mas pequeño es el objeto que aparece en la escena.

Children
hijos de la clase Axes (ejes). Un vector que contiene los manejadores de todos los objetos graficos dejados en los ejes (sean o no visibles). Los objetos graficos que no pueden ser hijos de la calse Axes son: images (imagenes), lights (luces), lines (lineas), patches (parches), surfaces (superficies) y text (texto).

CLim
Un vector del dos elemento [cmax cmin] que determina cómo MATLAB traza los valores de los datos de el colormap. cmin es el valor de los datos a trazar a la primera entrada del colormap, y el cmax es el valor de los datos a trazar a la ultima entrada del colormap.

Color
color de los planos de fondo de los ejes. Ajustando esta propiedad a *none* (no) quiere decir que los ejes son transparentes y el color de la figura se ven atraves de ellos. un *ColorSpec* (color especifico) es un vector de tres elementos RGB o un nombre predefinido en Matlab.

ColorOrder
Colores a usar para graficas multiples. Es una matriz de m por 3 de valores RGB que definen el color usado en las funciones plot y plot3 para colorear cada linea graficada.

CreateFcn
Rutina de llamada ejecutada cuando se crea un objeto. Esta propiedad define una rutina de llamada que es ejecutada cuando matlab crea un objeto de la clase Axes. Se debe definir esta propiedad como un valor por default para los ejes.

CurrentPoint
Posicion de el ultimo boton al que se le dio click, en unidades de datos de los ejes. Una matriz de 2 por 3 que contiene las coordenadas de dos puntos definidos por la locacion de el cursor. Estos dos puntos estan en la linea que es perpendicular al plano de la pantalla y pasa por el cursor.

DeleteFcn
Una rutina de llamada que se ejecuta cuando borras un objeto Axes. Matlab ejecuta la rutina antes de destruir las propiedades del objeto, asi la rutina de llamada puede buscar

sus valores

DrawMode
Modo de dibujo.

- *normal* - Dibuja los objetos de atrás hacia adelante respecto a la visión actual.
- *fast* - dibuja objetos en la orden en la cual usted la especificó originalmente. Esto inhabilita clasificar tridimensional realizado generalmente por MATLAB, dando por resultado más rápidamente la representación.

FontAngle
Inclinación de un Caracter. Poniendo esta propiedad en *Italic* (italica) o *oblique* (oblicua) selecciona una version inclinada de la fuente, cuando esta disponible en tu sistema.

FontName
El nombre de la fuente que mostrara la cadena. Para mostrar e imprimir correctamente, debe ser un tipo de fuente que tu sistema soporte.
Para usar un ancho ajustado que se vea bien en cualquier exterior (y que se muestre correctamente en Japon, donde usan caracteres "multibyte"), Pon al FontName la cadena FixedWidth (esta cadena es sencible a la mayusculas):

```
set( uicontrol_handle, 'FontName', 'FixedWidth' )
```

FontSize
Tamaño de la fuente. Un numero que especifica el tamaño de la fuente que va a ser mostrado en la cadena, in unidades determinadas por la propiedad *FontUnits*. El tamaño por default es dependiente del sistema.

FontUnits
Unidades del tamaño de la fuente. Esta propiedad determina las unidades usadas por la propiedad *FontSize*.Las unidades normalizadas interpretan el *FontSize* como una fraccion de la altura de el Axes.

FontWeight
Peso de un caracter. El peso de un caracter para el texto de los ejes. Poniendo esta propiedad en *Bold* hace que Matlab use una version "negrita" de la fuente, cuando esta disponible en tu sistema.

GrindLineStyle
Estilo de lineas usado para dibujar la cuadricula. El estilo de linea es una cadena que consiste en un caracter en cuotas, especificando una linea solida (-), guiones (--), lineas de puntos (:), o guiones con puntos(-.)

HandleVisibility
Controla el acceso al manejador (handle) de un objeto por usuarios de la linea de comando y GUI's. Esta propiedad determina cuando un manejador de un objeto es visible en la lista de los objetos hijos de su clase papa.
HandleVisibility es util para prevenir usuarios de la linea de comando de accidentalmente borrar o dibujar en una figura que contiene solo dispositivos de interfase de usuarios.

Los manejadores son siempre visibles cuando HandleVisibilty esta activada (en on).
Asignando HandleVisibility a una llamada hace que el manejador sea visible para rutinas de llamada o funciones invocadas po rutinas de llamada, pero no para las que son invocadas desde la linea de comando. Esto es para proteger los GUI's de los usuarios de la linea de comando, y tambien permite a las rutinas de llamada tener el completo acceso a los manejadores de los objetos.

Poniendo HandleVisibility en off hace al manejador siempre invisible. Esto puede ser necesario cuando una rutina de llamada invoca a una funcion que puede dañar al GUI, por que asi temporalmente esconde sus propios manejadores mientras se ejecuta dicha funcion.
Cuando un manejador no es visible en la lista de hijos de su clase papa, no puede ser regresado por funciones que obtebgan manejadores buscando la jerarquia del objeto o "preguntando" las propiedades del manejador. Esto incluye *get, findobj, gca, gcf, gco, newplot*, *cla, clf* y *close.*

Cuando la visibilidad del manejador (HandleVisibility) es restringida usando llamadas o poniendola en off, el manejador del objeto no aparece en la propiedad children de su papa, las figuras no aparecen en la propiedad *CurrentFigure* de *Root* (raiz), los objetos no aparecen en la propiedad de *Root CallbackObjet* o en la propiedad de la figura *CurrentObjet*, y los *Axes* (ejes) no aparecen en la propiedad *CurrentAxes* de sus padres.

Puedes poner la propiedad de Root *ShowHiddenHandles* en on para hacer visibles a todos los manejadores, sin importar los ajustes de su propiedad *HandleVisibility* (esto no afecta sus valores).

Los manejadores que estan escondidos siguen siendo validos, si se conoce el manejador de un objeto tu puedes asigner (set) y obtener (get) sus propiedades, y pasarcelas a cualquier funcion que opere manejadores

Interruptible
Rutina de llamada de modo interumpible. La característica interrumpible controla si una rutina de llamada de los ejes se puede interrumpir por rutinas de llamada posteriormente invocadas. Solamente las rutinas de llamada definidas para el ButtonDownFcn son afectadas por la característica interrumpible. Matlab comprueba para saber si hay

acontecimientos que puedan interrumpir una rutina de llamada solamente cuando encuentra un *drawnow,* una *figura,* un *getframe,* o un comando de la pausa en la rutina.

El fijar interrumpible a encendido permite que la rutina del servicio repetido de cualquier objeto de los gráficos interrumpa las rutinas del servicio repetido que originan de una característica de los ejes. Observe que Matlab no ahorra el estado de las variables o de la exhibición cuando ocurre una interrupción.

Layer
Dibuja las líneas de ejes debajo o sobre objetos gráficos. Esta característica determina si las lineas de los ejes y las marcas de la señal dibuja en arriba o debajo de los objetos de los hijos de los ejes para cualquier vista de 2-D. Esto es útil para poner líneas de rejilla y marcas de la señal encima de imágenes.

LineStyleOrder
Orden del estilo de la línea de los ejes. Una cadena que especifica estilos de la línea en el orden que se usan para trazar líneas múltiples en los ejes.

LineWidth
Ancho de las lineas de los ejes x, y, y z.
El ancho, en puntos, de las líneas que representan cada eje. La anchura de la línea de la por omisión es 0,5 puntos.

NextPlot
Manipulación de los ejes por graficas subsecuentes.

- new - Crea nuevos ejes antes de dibujar.
- add - Agrega nuevos obgetos a los ejes actuales.
- replace - Destruye los ejes actuales y crea nuevos

Parent
Identificador de el objeto papa de los ejes. Una propiedad de solo lectura que contiene el Identificador de el objeto papa de los ejes. El papa de un objeto tipo *Axes* es la figura en que es expuesta.

Position
Tamaño y posicion de los ejes. Un vector de cuatro elementos [left bottom width height], donde *left* (izquierda) y *bottom* (abajo) son la distancia de la esquina inferior izquierda de la ventana a la esquina inferior izquierda de los ejes. *width* (ancho) y *height* (altura) son las dimenciones de el rectangulo de los ejes. Todas las medidas estan definidas por la propiedad *Units* (unidades).

TickDir
La dirección de la marca de la señal
- *in* – las marcas de la señal se dirige hacia adentro de las líneas del eje
- *out* - las marcas de la señal se dirige hacia afuera de las líneas del eje

TickLength
Tamaño de la marca de la señal. Un vector de dos elementos [2Dlength 3Dlength] que especifica el tamaño de la marca de la señal. Donde:
2Dlength es la longitud de las marcas de la señal usadas para las vistas de dos dimensiones.
3Dlength es la longitud de las marcas de la señal usadas para las vistas de dos dimensiones.

Title
Titulo de los ejes. El identificador de el objeto de texto que define el titulo de los ejes. Primero crea el objeto de texto para despues obtener su manejador. La siguiente declaracion ejecuta ambos pasos:

```
set(gca,'title',text(0,0,'axes title'))
```

Type
Tipo de objetos graficos. Esta propiedad siempre es "axes" para los objetos *axes* (ejes).

Units
Unidades de medida. Las unidades que Matlab usa para interpretar la propiedad Position. Todas las unidades son medidas de la esquina inferior izquierda de la figura ventana.

UserData
Datos especificados por el usuario. Cualquier matriz que quieras asociar con un objeto de tipo axes. El objeto no usa este dato pero tu puedes recuperarlo usando el comando get

View
Punto de vista de los ejes. Un vector de dos elementos, [az, el], que establece el punto de vista usado para transformar graficas de tercera dimension en un espacion bi dimencional de la pantalla. El viewpoint (punto de vista) es la pocision de la camara que obserba la grafica de 3-D, donde *az* es el acimut. El acimut gira alrededor del eje z, con valores positivos haceque el punto de vista ruede en un contador en el sentido de las manecillas del reloj.
el es la elevacion, y especifica el angulo sobre o bajo el objeto. Valores positivos hacen que el punto de vista se mueva arriba de el objeto. Especifica cada elemento en grados.

Visible
Visibilidad de los ejes. Por defecto, todos los objetos Axes son visibles. Cuando esta propiedad
propiedad esta desactivada (en off), los ejesl no es visible, pero sigue existiendo y puedas buscar y modificar su propiedades.

XForm
Una matriz de 4X4 que define la transformacion de graficas tridimencionales a la pantalla bidimencional.

- Propiedades que controlan el eje X -

Xcolor
Color de el eje X. Un vector de tres elementos RGB, o una cadena predefinida por matlab. Esta propiedad determina el color de la exhibición para el eje *x* ,las marcas de la señal,las etiquetas de la marca de la señal, y las lineas de las rejillas.

Xdir
Dirección de aumentar los valores de x.

- *normal* - Los valores de *x* aumentan de izquierda a derecha (sistema coordinado derecho).
- *reverse* - los valores de *x* aumentan de derecho a la izquierda.

Xgrid
Modo de linea de rejilla del eje X.

- on - Matlab dibuja lineas perpendiculares al eje X en cada marca de la señal
- off - No dibuja lineas

Xlabel
El identificador del objeto de texto usado para etiquetar el eje X. Primero cree el objeto del texto para después obtener el identificador. La declaración siguiente realiza ambos pasos:

```
set(gca,'xlabel',text(0,0,'axis label'))
```

Mientras que la funcion Text requiere la posicion del dato, esto no es usado para acomodar el texto. En lugar, Matlab pone la cadena ‘axis label’ a la posición correcta para el eje X.

Xlim
Límites del eje X. Un vector del dos-elemento [xmax xmin] que especifica los valores mínimos y máximos del eje X, donde el xmin es el valor mínimo del eje X. Y xmax es el valor máximo. El valor por defecto para XLim es [0 1].

XlimMode
Modo de los límites del eje X.

- *auto* - Matlab calcula los límites del eje X (XLim) que palmo el XData de los hijos de los ejes y produce números redondeados para los límites de el eje X
- *manual* - Matlab tomas de los límites del eje X de XLim; los límites no dependen del XData en los objetos hijos. Fijar los valores para XLim pone esta característica en manual.

Xscale
Escala de los ejes X.

- Linear – Escala linear
- Log – Escala logarítmica

Xtick
Espaciamiento de la marca de la señal del eje X. Un vector de los valores que corresponden a los valores de los datos X en cuál usted desea poner marcas de la señal. Si usted no desea las marcas de la señal exhibidas, Asigne XTick el vector vacío, [ ].

XtickLabel
Etiquetas de la marca de la señal del eje X. Una matriz de las cadenas a utilizar como etiquetas para las marcas de la señal a lo largo del eje X. Estas etiquetas sustituyen las etiquetas numéricas generadas por MATLAB. Si usted no especifica suficientes etiquetas del texto para todas las marcas de la señal, MATLAB utiliza todas las etiquetas especificadas, y después etiqueta las marcas restantes de la señal reutilizando las etiquetas especificadas. El comando

```
set(gca,'XTickLabels',['Old Data';'New Data'])
```

etiqueta las primeras dos marcas de la señal en el eje X *Old Data* (datos viejos) y *New Data* (nuevos datos) respectivamente. Cada cadena de caracteres debe tener un número igual de caracteres debido a la manera que MATLAB almacena las cadenas. Esta característica no controla el número de las marcas de la señal o sus localizaciones.

XtickLabelMode
Modo de etiquetas de la marca de la señal del eje X

- *auto* - MATLAB calcula las etiquetas de la señal del eje X (XTickLabels) que expande el dato X de los hijos de los ejes.

- manual - MATLAB toma etiquetas de la señal del eje X de *XTickLabels*; no depende del dato X en los objetos de los niños. asignarle valores a *XTickLabels* fija esta característica en manual.

XtickMode
Modo de la marca de la señal del eje X.

- auto – Matlab calcula el espaciamiento de la marca de la señal del eje X (XTick) que expande el dato X de los hijos de los ejes.
- Manual – Matlab toma el espaciamiento de la marca de el eje X de Xtick; no depende de el dato X de los objetos hijos. Fijar los valores para XTick fija esta característica al manual.

- Propiedades que controlan el eje Y -

Ycolor
Color de el eje Y. Un vector de tres elementos RGB, o una cadena predefinida por matlab. Esta propiedad determina el color de la exhibición para el eje *x* ,las marcas de la señal,las etiquetas de la marca de la señal, y las lineas de las rejillas.

Ydir
Dirección de aumentar los valores de x.

- *normal* - Los valores de *x* aumentan de izquierda a derecha (sistema coordinado derecho).
- *reverse* - los valores de *x* aumentan de derecho a la izquierda.

Ygrid
Modo de linea de rejilla del eje Y.

- on - Matlab dibuja lineas perpendiculares al eje Y en cada marca de la señal
- off - No dibuja lineas

Ylabel
El identificador del objeto de texto usado para etiquetar el eje Y. Primero cree el objeto del texto para después obtener el identificador. La declaración siguiente realiza ambos pasos:

```
set(gca,'xlabel',text(0,0,'axis label'))
```

Mientras que la funcion Text requiere la posicion del dato, esto no es usado para acomodar el texto. En lugar, Matlab pone la cadena ‘axis label’ a la posición correcta para el eje Y.

Ylim
Límites del eje Y. Un vector del dos-elemento [xmax xmin] que especifica los valores mínimos y máximos del eje Y, donde el xmin es el valor mínimo del eje Y. Y xmax es el valor máximo. El valor por defecto para YLim es [0 1].

YlimMode
Modo de los límites del eje Y.

- *auto* - Matlab calcula los límites del eje Y (YLim) que palmo el YData de los hijos de los ejes y produce números redondeados para los límites de el eje Y
- *manual* - Matlab tomas de los límites del eje Y de YLim; los límites no dependen del YData en los objetos hijos. Fijar los valores para YLim pone esta característica en manual.

Yscale
Escala de los ejes Y.

- Linear – Escala linear
- Log – Escala logarítmica

Ytick
Espaciamiento de la marca de la señal del eje Y. Un vector de los valores que corresponden a los valores de los datos Y en cuál usted desea poner marcas de la señal. Si usted no desea las marcas de la señal exhibidas, Asigne YTick el vector vacío, [ ].

YtickLabel
Etiquetas de la marca de la señal del eje Y. Una matriz de las cadenas a utilizar como etiquetas para las marcas de la señal a lo largo del eje Y. Estas etiquetas sustituyen las etiquetas numéricas generadas por MATLAB. Si usted no especifica suficientes etiquetas del texto para todas las marcas de la señal, MATLAB utiliza todas las etiquetas especificadas, y después etiqueta las marcas restantes de la señal reutilizando las etiquetas especificadas. El comando

```
set(gca,'YTickLabels',['Old Data';'New Data'])
```

etiqueta las primeras dos marcas de la señal en el eje Y *Old Data* (datos viejos) y *New Data* (nuevos datos) respectivamente. Cada cadena de caracteres debe tener un número igual de caracteres debido a la manera que MATLAB almacena las cadenas. Esta característica no controla el número de las marcas de la señal o sus localizaciones.

YtickLabelMode
Modo de etiquetas de la marca de la señal del eje Y

- *auto* - MATLAB calcula las etiquetas de la señal del eje Y (YTickLabels) que expande el dato Y de los hijos de los ejes.
- manual - MATLAB toma etiquetas de la señal del eje Y de *YTickLabels*; no depende del dato Y en los objetos de los niños. asignarle valores a *YTickLabels* fija esta característica en manual.

YtickMode
Modo de la marca de la señal del eje Y.

- auto – Matlab calcula el espaciamiento de la marca de la señal del eje Y (YTick) que expande el dato Y de los hijos de los ejes.
- Manual – Matlab toma el espaciamiento de la marca de el eje Y de Ytick; no depende de el dato Y de los objetos hijos. Fijar los valores para YTick fija esta característica al manual.

- Propiedades que controlan el eje Z -

Zcolor
Color de el eje Z. Un vector de tres elementos RGB, o una cadena predefinida por matlab. Esta propiedad determina el color de la exhibición para el eje *x* ,las marcas de la señal,las etiquetas de la marca de la señal, y las lineas de las rejillas.

Zdir
Dirección de aumentar los valores de x.

- *normal* - Los valores de *x* aumentan de izquierda a derecha (sistema coordinado derecho).
- *reverse* - los valores de *x* aumentan de derecho a la izquierda.

Zgrid
Modo de linea de rejilla del eje Z.

- on - Matlab dibuja lineas perpendiculares al eje Z en cada marca de la señal
- off - No dibuja lineas

Zlabel
El identificador del objeto de texto usado para etiquetar el eje Z. Primero cree el objeto del texto para después obtener el identificador. La declaración siguiente realiza ambos pasos:

```
set(gca,'xlabel',text(0,0,'axis label'))
```

Mientras que la funcion Text requiere la posicion del dato, esto no es usado para acomodar el texto. En lugar, Matlab pone la cadena ‘axis label’ a la posición correcta para el eje Z.

Zlim
Límites del eje Z. Un vector del dos-elemento [xmax xmin] que especifica los valores mínimos y máximos del eje Z, donde el xmin es el valor mínimo del eje Z. Y xmax es el valor máximo. El valor por defecto para ZLim es [0 1].

ZlimMode
Modo de los límites del eje Z.

- *auto* - Matlab calcula los límites del eje Z (ZLim) que palmo el ZData de los hijos de los ejes y produce números redondeados para los límites de el eje Z
- *manual* - Matlab tomas de los límites del eje Z de ZLim; los límites no dependen del ZData en los objetos hijos. Fijar los valores para ZLim pone esta característica en manual.

Zscale
Escala de los ejes Z.

- Linear – Escala linear
- Log – Escala logarítmica

Ztick
Espaciamiento de la marca de la señal del eje Z. Un vector de los valores que corresponden a los valores de los datos Z en cuál usted desea poner marcas de la señal. Si usted no desea las marcas de la señal exhibidas, Asigne ZTick el vector vacío, [ ].

ZtickLabel
Etiquetas de la marca de la señal del eje Z. Una matriz de las cadenas a utilizar como etiquetas para las marcas de la señal a lo largo del eje Z. Estas etiquetas sustituyen las etiquetas numéricas generadas por MATLAB. Si usted no especifica suficientes etiquetas del texto para todas las marcas de la señal, MATLAB utiliza todas las etiquetas especificadas, y después etiqueta las marcas restantes de la señal reutilizando las etiquetas especificadas. El comando

```
set(gca,'ZTickLabels',['Old Data';'New Data'])
```

etiqueta las primeras dos marcas de la señal en el eje Z *Old Data* (datos viejos) y *New Data* (nuevos datos) respectivamente. Cada cadena de caracteres debe tener un número igual de caracteres debido a la manera que MATLAB almacena las cadenas. Esta característica no controla el número de las marcas de la señal o sus localizaciones.

ZtickLabelMode
Modo de etiquetas de la marca de la señal del eje Z

- *auto* - MATLAB calcula las etiquetas de la señal del eje Z (ZTickLabels) que expande el dato Z de los hijos de los ejes.
- manual - MATLAB toma etiquetas de la señal del eje Z de *ZTickLabels*; no depende del dato Z en los objetos de los niños. asignarle valores a *ZTickLabels* fija esta característica en manual.

ZtickMode

Modo de la marca de la señal del eje Z.

- auto – Matlab calcula el espaciamiento de la marca de la señal del eje Z (ZTick) que expande el dato Z de los hijos de los ejes.
- Manual – Matlab toma el espaciamiento de la marca de el eje Z de Ztick; no depende de el dato Z de los objetos hijos. Fijar los valores para ZTick fija esta característica al manual.